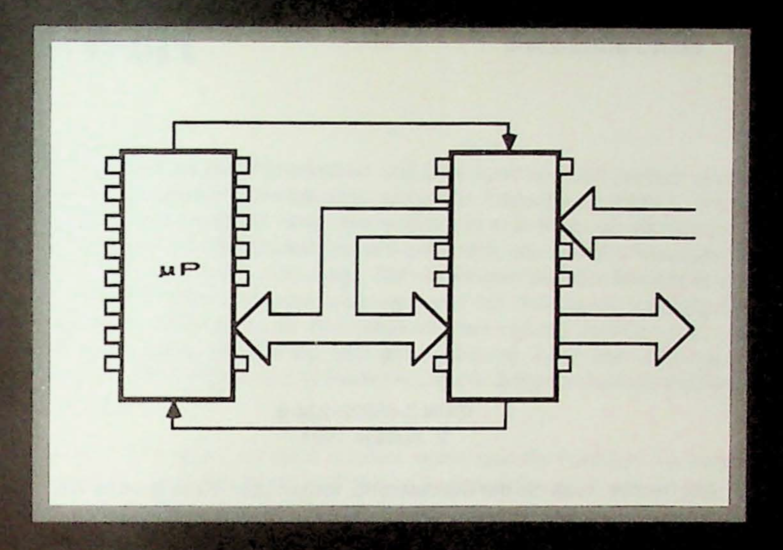
Das COMPUTER-PERIPHERIE Kochbuch

Andreas Roth





Das COMPUTER-PERIPHERIE Kochbuch

Andreas Roth



CIP-Titelaufnahme der Deutschen Bibliothek

Roth, Andreas:
Das Computer-Peripherie-Kochbuch / Andreas Roth. – 2. Aufl. –
Vaterstetten bei München : IWT-Verl., 1991
ISBN 3-88322-234-8

ISBN 3-88322-234-8
2. Auflage 1991

Herstellung: Freiburger Graphische Betriebe, Freiburg
Umschlaggestaltung: CommunAction, München

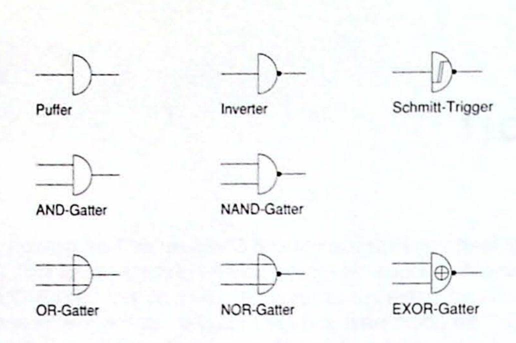
Vorwort

Über die Fähigkeiten von Prozessoren und Computer wird viel geredet und in der Tat ist ihr Leistungsvermögen bei entsprechender Software beachtlich. Doch würden sie alle Aufgaben selbst verrichten wollen, die man heute an ein Computersystem stellt, wären sie überfordert und ohne die Hilfe von Peripheriebausteinen zu gewissen Dingen gar nicht in der Lage. Daher reduziert sich die Aktivität eines Prozessors auf das Treffen von Entscheidungen und das Delegieren von Aufgaben an die Peripheriebausteine, die für ihre spezielle Verwendung optimiert sind und die übertragene Aufgabe selbständig und ohne weiteres Zutun der CPU erledigen. Das äußert sich in Zeitgewinn und Reduzierung der Software-Komplexität und des Software-Umfangs.

Diese Vorteile kann man nur dann nutzten, wenn man die Funktion der Bausteine genau kennt. Daher konzentriert sich dieses Buch auf die detaillierte Beschreibung der Hard- und Software-Eigenschaften einiger ausgewählter Peripheriebausteine. Die Auswahl ist dabei so getroffen, daß mit deren Hilfe der Aufbau eines modernen Computersystems in vielen Variationen möglich ist. Es finden sich ausschließlich Bausteine, die zu den Prozessoren der 80XX-Reihe kompatibel sind, d.h. die über die gleiche Busstruktur verfügen. Dazu gehören Bausteine, die als Standard in fast jedem System zu finden sind und solche, die mit hohem Integrationsgrad den Datenumsatz wesentlich beschleunigen und die Hardware erheblich reduzieren.

Je komplexer ein Peripheriebaustein ist, desto vielfältiger sind seine Funktionen und seine Programmierbarkeit. Im gleichen Maße wächst der Umfang, der zur Beschreibung der vollständigen Möglichkeiten des Bausteins nötig ist.

Obwohl durch die Verwendung hochintegrierter Bausteine die Zahl der externen Logik-Schaltkreise verschwindend gering ist, werden in den Abbildungen gelegentlich welche benutzt. Dabei ist die Zuordnung von Symbol und logischer Struktur nach folgender Abbildung gegeben. Logische Funktionen sind im Text durch eine *kursive Schrift* gekennzeichnet.



Die Bezeichnungen, Abkürzungen und Mnemoniks der Hersteller werden in Anpassung an die internationale Nomenklatur unverändert benutzt. Für Fachausdrücke wird nur dann ein deutscher Begriff verwendet, wenn er im Sprachgebrauch ein eindeutiges Verständnis zuläßt. In zahlreichen Fällen ist eine Übersetzung aus dem Englischen nicht sinnvoll, da der Begriff einen Terminus technicus darstellt. Solche auftretenden Fremdwörter sind im Text definiert, um dessen Verständnis zu gewährleisten.

Ein $\overline{\text{Strich}}$ über eine Signalbezeichnung kennzeichnet ein low-aktives Signal. Ein typisches Beispiel ist die $\overline{\text{CS}}$-Leitung, die den betreffenden Baustein genau dann für die Kommunikation mit der CPU freigibt, wenn die Leitung Low-Pegel (Masse) führt.

Ein besonderer Dank für die freundliche Unterstützung gilt den Firmen Mitsubishi, National Semiconductor, Harris, Intel, Matra Harris Semiconductor und Hitachi.

Andreas Roth

Inhaltsverzeichnis

Kapitel 2 2-1

Kapitel 3 3-1

Kapitel 1

1.1 Adreß- und Datenbuskonzepte

Um die Zahl der Anschlüsse am Prozessorgehäuse niedrig zu halten, geben die Mehrzahl der Prozessoren und Mikrocontroller die Adressen und Daten teilweise über dieselben Pins aus. Das kann natürlich nicht zu derselben Zeit erfolgen, da sich die logischen Pegel der Daten und Adressen stören würden. Daher erfolgt die Ausgabe von Adressen und Daten nacheinander oder gemultiplext, wie der Fachausdruck lautet.

Um Daten in eine Speicherstelle zu schreiben, gibt der Prozessor zuerst (Phase 1, Bild 1-1) an den gemultiplexten Adreß-/Datenleitungen, die mit AD bezeichnet werden, die Adresse aus, gefolgt von einem Signal, mit dessen Hilfe die Adresse in einem oder mehreren externen Adreßzwischenspeicher aufgefangen wird. Das Signal heißt Address Latch Enable und wird mit ALE bezeichnet. Alle Prozessoren mit gemultiplextem Adreß-/Datenbus erkennt man an dem Vorhandensein eines ALE-Anschlusses. Da die Hauptzahl der Systemspeicher und Peripheriebausteine während eines Schreib-/Lesezugriffs eine stabile Adresse erfordern, gibt der Adreßzwischenspeicher die nunmehr gemultiplexte Adresse an die Bausteine aus. Häufig ist dem Zwischenspeicher ein Adreßdekoder nachgeschaltet, der einen Peripheriebaustein oder einen Speicherbereich aus vielen auswählt. Während dieses Vorgangs steht natürlich auch die Adresse an den Eingängen des Speichers oder Peripherie an. Doch da die beiden Steuersignale nicht aktiv sind, sind die Pins der Peripheriebausteine im Tri-State und nehmen somit weder Daten auf, noch geben sie welche ab.

Werden nun Daten vom Prozessor in den Speicher oder Peripherie geschrieben (Phase 2, Bild 1-2), erscheinen sie an den Pins AD und liegen gleichzeitig am Eingang des Adreßzwischenspeichers und der Peripherie an. Danach gibt der Prozessor ein Schreibsignal $\overline{WR}$ aus, das nur auf den Peripheriebaustein, nicht aber auf den Adreßzwischenspeicher wirkt. Dabei werden die auf dem Datenbus liegenden Daten in die Peripherie geschrieben.

Das Einlesen von Daten oder Befehlskodes durch den Prozessor geschieht in ähnlicher Weise wie das Schreiben (Phase 2). Der Speicher oder die Peripherie ist nach Phase 1 adressiert und die Pins AD des Prozessors sind als Eingänge konfiguriert, häufig im Tri-State. Mit einem Lesesignal $\overline{RD}$ wird der Speicher oder die Peripherie veranlaßt, die Daten auf den Datenbus zu legen (Bild 1-3).

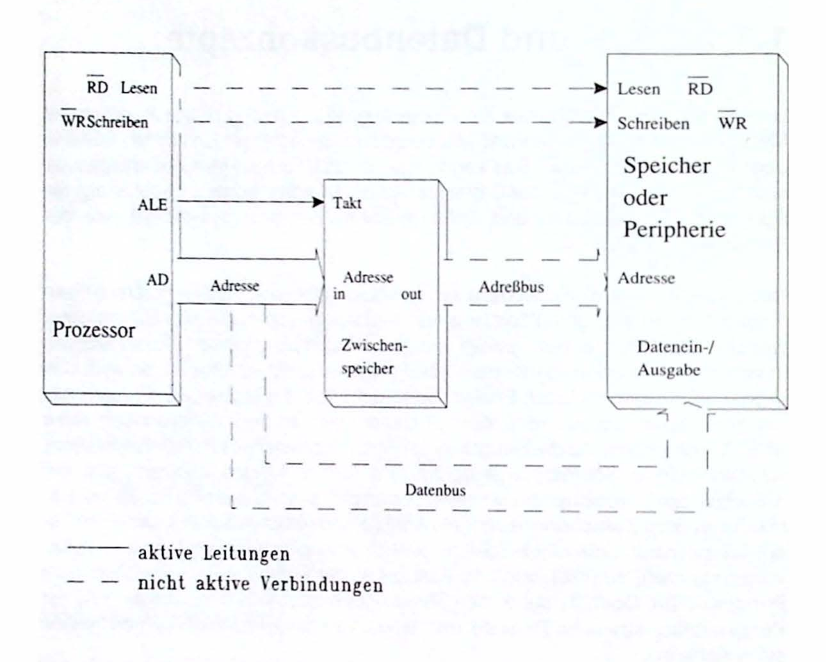


Bild 1-1. Aktive Leitungen bei der Adreßausgabe und Zwischenspeicherung

RD Lesen
WR Schreiben
ALE
AD
Prozessor
Daten
Takt
Adresse
in
out
Zwischen-
speicher
Adreßbus
Lesen RD
Schreiben WR
Speicher
oder
Peripherie
Adresse
Datenein-/
Ausgabe
Datenbus

——— aktive Leitungen
— — nicht aktive Verbindungen

Bild 1-2. Aktive Leitungen beim Schreiben von Daten

RD Lesen
WR Schreiben
ALE
AD
Prozessor
Daten
Takt
Adresse
in
out
Zwischen-
speicher
Adreßbus
Lesen RD
Schreiben WR
Speicher
oder
Peripherie
Adresse
Datenein-/
Ausgabe
Datenbus

——— aktive Leitungen
— — nicht aktive Verbindungen

Bild 1-3 Aktive Leitungen beim Lesen von Daten

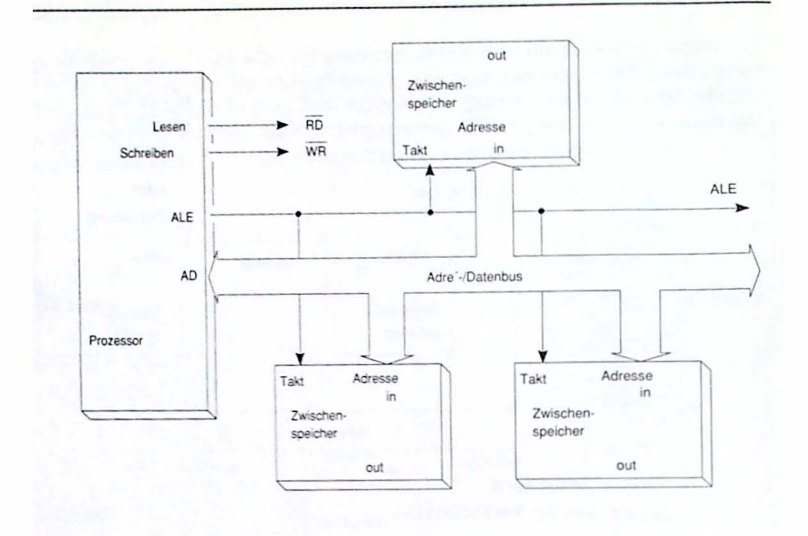


Bild 1-4. Gemultiplexter Bus mit lokaler Demultiplexung

Für die Adreßzwischenspeicherung werden statische 8-Bit-Speicher mit der Bezeichnung 8282, 8283, 74HC373 etc. verwendet. Für eine 20-Bit-Adresse benötigt man somit mindestens drei Bausteine. Die Speicher sind meist nicht invertierend (8282, 74HC373) und die Ausgänge weisen dieselben Pegel wie die Eingänge auf. An den Ausgängen finden sich Tri-State-Verstärker, die einen Strom von 32 mA zur Verfügung stellen. Sie können eine 300 pF-Last in 30 ns schalten. Die Benutzung invertierender Speicher (8283) stellt im Prinzip keinen Nachteil für das System dar, wenngleich nun der Speicher physikalisch rückwärts, also von FFFF nach 0000 adressiert wird. Für den Prozessor ist dieser Umstand nicht relevant. Bei einer Systementwicklung verlangt die Invertierung besondere Aufmerksamkeit, da nicht nur die Adreßdekodierlogik, sondern auch der Einsatz von häufig mitgespeicherten Steuersignalen besonders aufmerksam geplant werden muß. Der Vorteil kann in der Reduzierung notwendiger Logikschaltungen liegen. Der 8283 liefert einen Strom von 32 mA, schaltet aber eine Last von 300 pF in bereits 22 ns.

Die Daten am Eingang werden erst mit der negativen Flanke des ALE-Signals in die Bausteine übernommen. Damit die Ausgänge aktiv werden können, muß der Output-Enable-Pin $\overline{OE}$ an Masse gelegt werden. Das Demultiplexen des gemultiplexten Adreß-/Datenbus, d.h. Speichern der Adresse von dem gemultiplexten Bus, kann an verschiedenen Orten des Systems (Bild 1-4) oder gleich bei der CPU mit Aufspaltung in Adreß- und Datenbus für alle Bausteine des Systems (Bild 1-1

bis 1-3) erfolgen. Die am häufigsten angewandte Technik, auf die im folgenden immer wieder zurückgegriffen wird, ist die Trennung von Adressen und Daten gleich nach Ausgabe durch die CPU.

Während die Breite des Adreßbusses von der Art des Prozessors und des zur Verfügung stehenden Speicherplatzes bestimmt ist, schwankt die Breite des Datenbusses je nach Prozessor in Byte-Schritten. Damit ist die Zahl der Datenbits gemeint, die der Prozessor parallel ausgeben bzw. einlesen kann. Üblich sind Busbreiten von 8, 16 und 32 Bits. Es ist ferner üblich, eine Bitbreite von acht Bits als ein Byte zu bezeichnen, zwei Bytes als ein Wort und mehr als zwei Bytes als String . Das Byte eines Worts, das an der niedrigeren Speicherstelle steht (gerade Adresse), wird als Lowest Significant Byte (LSB) bezeichnet, und das an der höheren Adresse (ungerade Adresse) stehende wird als Most Significant Byte (MSB) bezeichnet. (In einem anderen Zusammenhang können die Abkürzungen LSB und MSB sich auch auf die Stellungen der Bits in einem Byte beziehen:
LSB = Bit 0; MSB = Bit 7).

Der Prozessor 8088 sieht nach außen wie ein 8-Bit-Prozessor aus, er verarbeitet intern jedoch die Daten mit einer 16-Bit-Breite. Dazu muß er natürlich zweimal auf den Speicher zugreifen, was den Datendurchsatz im Gegensatz zum 8086er, der über einen 16-Bit-Datenbus verfügt, etwas verlangsamt. Für manche Operationen benötigen 16-Bit-Prozessoren nur ein Byte. Zu diesem Zweck verfügen sie über ein Steuersignal (High Byte Enable $\overline{BHE}$), mit dessen Hilfe sie in Verbindung mit der Adreßleitung A_0 gezielt ein Byte eines Worts auswählen können. Die Verwendung beider Signale zeigt Tabelle 1-1 und Bild 1-5.

$\overline{BHE}$	A_0	Zugriff auf:
0	0	Ganzes Wort
0	1	MSB
1	0	LSB
1	1	Keine Funktion

Tabelle 1-1. Byte-Adressierung in einem 16-Bit-Datenbus

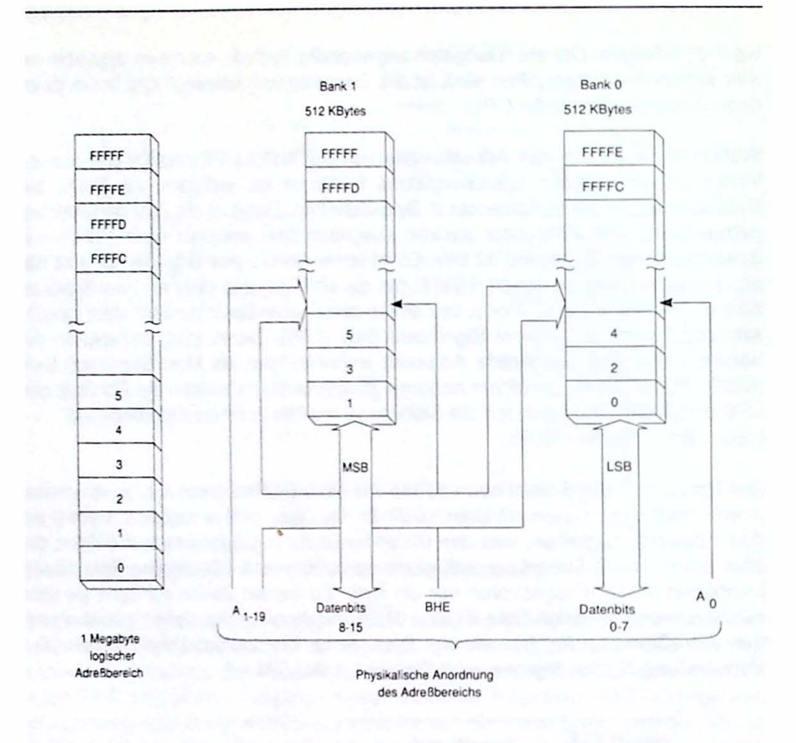


Bild 1-5. 1 MByte-Adreßbereich in einem 16-Bit-Datenbus

In Bild 1-5 ist ein Speicherbereich von 1 MByte mit einem 16 Bit breitem Datenbus zu sehen. 1 MByte entspricht der nicht verwendeten Einheit von ½ MWord. Der Speicherbereich wird durch die Signale $\overline{BHE}$ und A_0 in zwei Bänke mit je ½ MByte Kapazität und je einem 8-Bit-Datenbus unterteilt. Da dem Speicher nur die Adreßbits A_{1-19} zugeführt werden, erfolgt die Numerierung der Bytes in jeder Bank in zweier Schritten. Die Selektierung der Bänke erfolgt durch die Signale $\overline{BHE}$ und A_0. Streng genommen ist eine Auswahl der Bänke durch die beiden Signale nur beim Schreiben notwendig, da beim Schreiben von nur einem Byte die Daten in der nicht adressierten Bank vernichtet würden. Da das $\overline{BHE}$-Signal ein gemultiplextes Signal ist und ein gleiches Zeitverhalten wie die Adresse aufweist, sollte es auch mit dem ALE-Signal gespeichert werden, damit es stabil für einen Zugriff auf den Speicher oder Peripheriebaustein anliegt.

Wenn man vom System-Datenbus spricht, muß man zwei Unterscheidungen treffen:

1. Ein Datenbus, der neben Daten auch noch die gemultiplexte Adreßinformation aufweist (Bild 1-1).
2. Ein Datenbus, bei dem die Daten durch Transceiver von den Adressen getrennt sind (Bild 1-6).

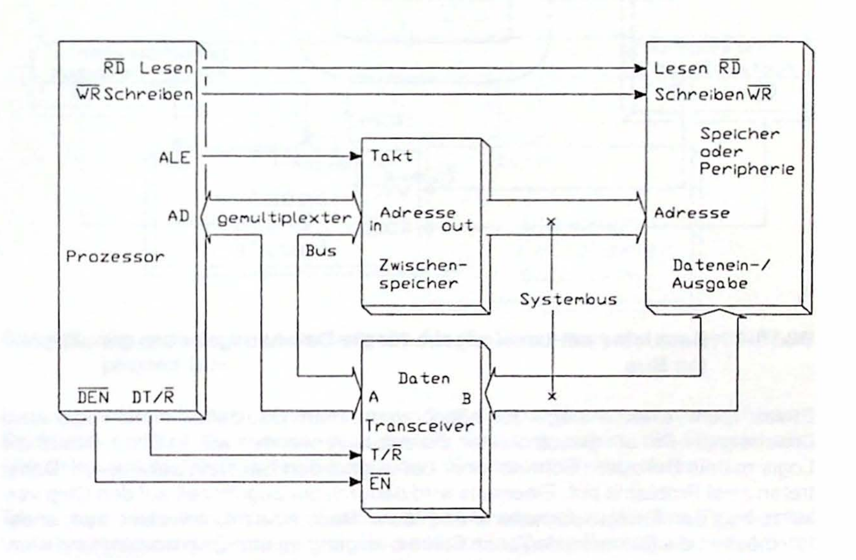


Bild 1-6. Gepufferter Datenbus

Wenn die Speicher oder Peripheriebausteine direkt mit gemultiplexten Bus in Verbindung stehen, muß man dafür Sorge tragen, daß sie durch eine eventuelle verzögerte Datenausgabe die Adreßverwaltung nicht stören. Sie müssen über einen Chip-Select-Eingang verfügen, mit dessen Hilfe sie ein- bzw. ausgeschaltet werden können. Auch sollten sie über eine Datenausgabesteuerung verfügen, über die die CPU auch bei selektiertem Baustein die Ausgänge zum Datenbus abschalten kann (Bild 1-7).

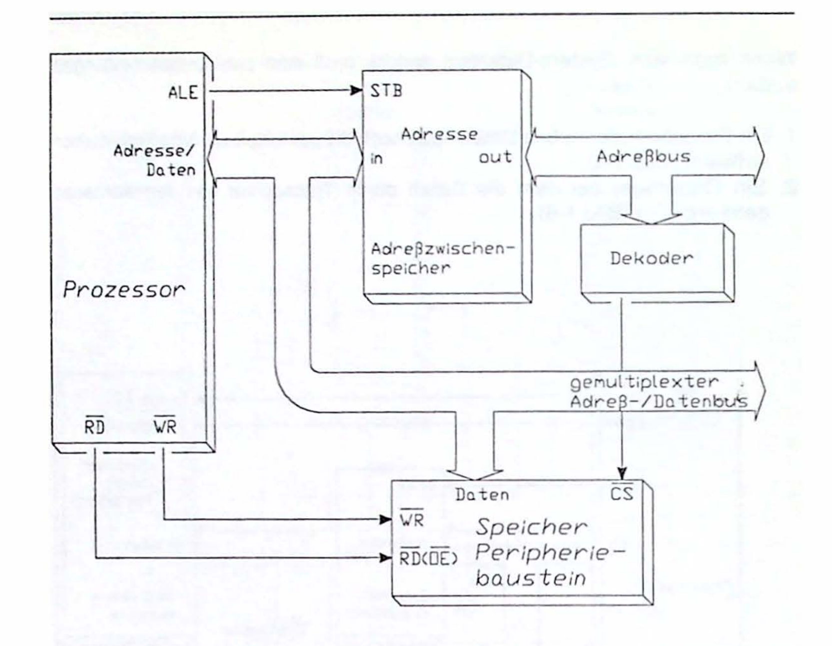


Bild 1-7. Bausteine mit Lesefreigabe für die Datenausgabe am gemultiplexten Bus

Etwas mehr externe Logik ist nötig, wenn man Bausteine ohne separatem Lesefreigabe-Pin am gemultiplexten Datenbus verwenden will. In Bild 1-8 muß die Logik mittels Dekoder-, Schreib- oder Lesesignal den Baustein selektieren. Dabei treten zwei Probleme auf. Einerseits wird dadurch die Zugriffszeit auf den Chip verkürzt, was den Einsatz schnellerer und teurer Bausteine mit sich zieht, zum anderen müssen die Einstellzeiten beim Schreibvorgang für den Chip ausreichend sein.

Die Verwendung des gemultiplexten Busses findet eine weitere Begrenzung in der Tatsache, daß fast alle Prozessoren nur einen auf 2,0 mA begrenzten Strom für eine Buskapazität von 100 pF zur Verfügung stellen. Gewöhnlich weisen Peripheriebausteine eine Kapazität von 20 pF , Adreßzwischenspeicher 12 pF und Speicherbausteine 1-12 pF pro Pin auf. Dann erschöpfen bereits drei Peripheriebausteine und zwei bis vier Speicherbausteine die Treiberfähigkeit des Prozessors.

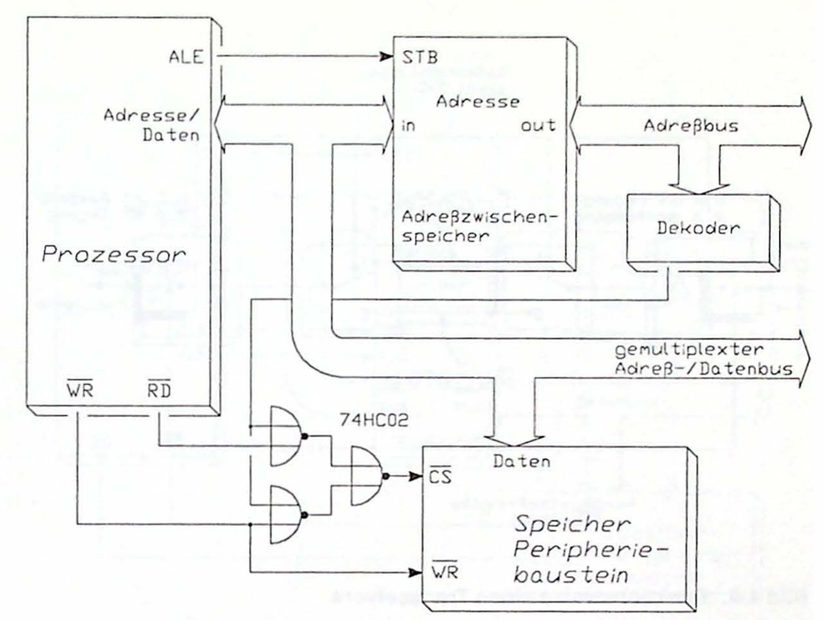


Bild 1-8. Bausteine ohne Lesefreigabe für die Datenausgabe am gemultiplexten Bus

Um dem erforderlichen Treiberstrom und der kapazitiven Last von größeren Systemen gerecht zu werden, müssen die Daten gepuffert, die Signale also verstärkt werden. Man verwendet dazu den Einsatz von Transceivern.

Transceiver ist ein Kunstwort, das aus den zwei englischen Wörtern **Trans**mitter und Re**ceiver** hervorgegangen ist. Es bedeutet, daß der Baustein aus der Sicht der CPU zum einem die Daten an die Peripherie überträgt und zum anderen die Daten empfängt. Die Übertragungsrichtung muß natürlich mit den Schreib-/Lesesignalen gekoppelt sein und wird mit dem Eingang T/R̄ (manchmal auch mit DIR bezeichnet) gesteuert. Darüber hinaus muß es möglich sein, die Übertragung, gleich in welcher Richtung, gänzlich zu unterbinden. Das muß zu dem Zeitpunkt erfolgen, an dem der Prozessor die Adresse ausgibt, die von den Zwischenspeichern aufgefangen werden soll. Für diesen Zweck besitzen manche Prozessoren einen Datenfreigabe-Pin (Data Enable, DEN), der den Transceiver dann einschaltet, wenn die Adresse auf dem Bus verschwunden ist und Daten fließen sollen. Eine symbolische Schaltung des Transceivers für ein Bit zeigt Bild 1-9. Die meisten der heute gebräuchlichen Transceiver umfassen eine Breite von acht Bits.

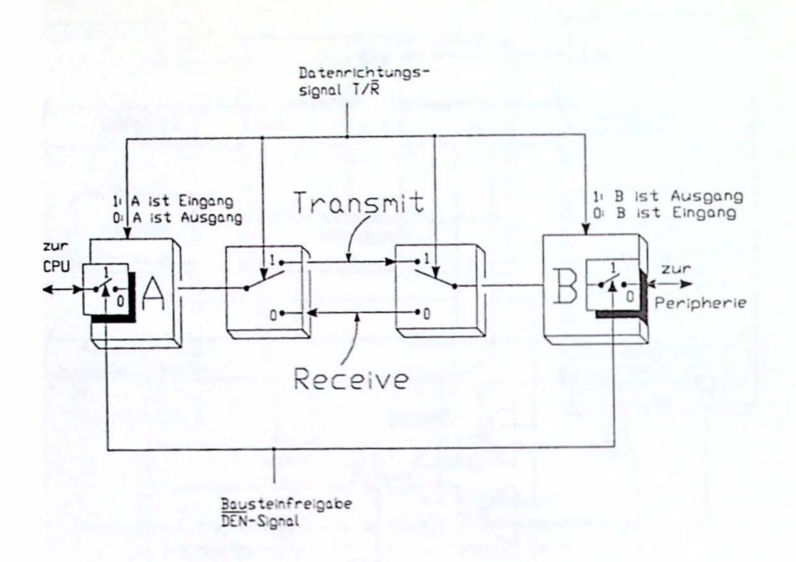


Bild 1-9. Funktionsweise eines Transceivers

Die Aufgabe des Transceivers erschöpft sich nicht nur in seiner Schalteigenschaft. Er ist vielmehr so gebaut, daß seine Ausgänge einen hohen Strom liefern können und über eine hohe Schaltgeschwindigkeit verfügen, um eine große Zahl von Bausteinen und die Kapazitäten langer Leitungen treiben zu können. So verfügen der nicht invertierende Transceiver 8286 und der invertierende 8287 über einen Strom von 32 mA in Richtung Peripherie/Speicher und in Richtung CPU über 10 mA. Sie können ein kapazitive Last von 300 pF in Richtung Peripherie/Speicher und eine von 100 pF in Richtung CPU innerhalb 22 ns (8287) oder 30 ns (8286) schalten. An Stelle des Bausteins 8286 bzw. 8287 können die Bausteine 74HC245 bzw. 74HC640 Verwendung finden. Sie sind preiswerter, haben die gleichen elektrischen Eigenschaften, sind funktionskompatibel, aber nicht pinkompatibel.

Auch wenn durch den Transceiver der Prozessor von den Speichern und Peripheriebausteinen getrennt ist, sind die Verbindungen der Steuersignale ähnlich wie beim gemultiplexten Adreß-/Datenbus. Die Unterschiede und Besonderheiten, die sich hierbei ergeben, zeigt Bild 1-11.

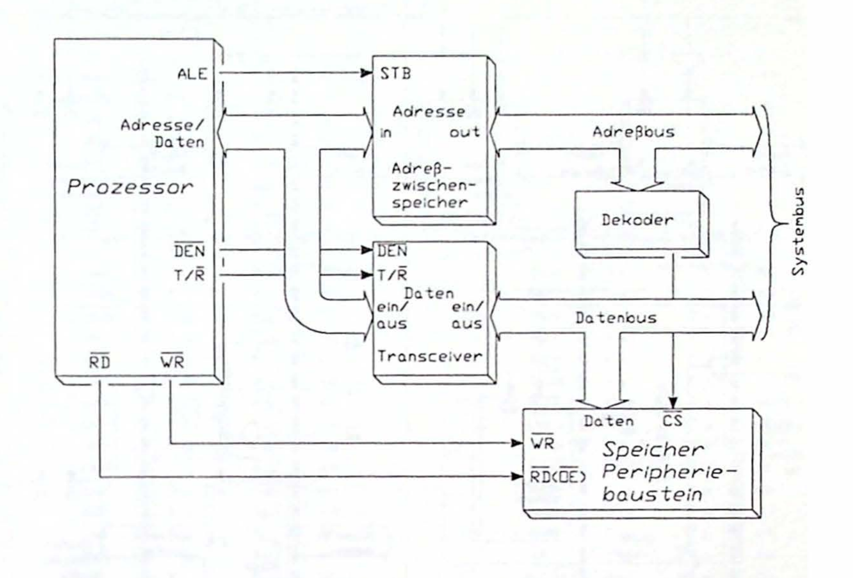


Bild 1-10. Bausteine mit Lesefreigabe für die Datenausgabe am System-Bus

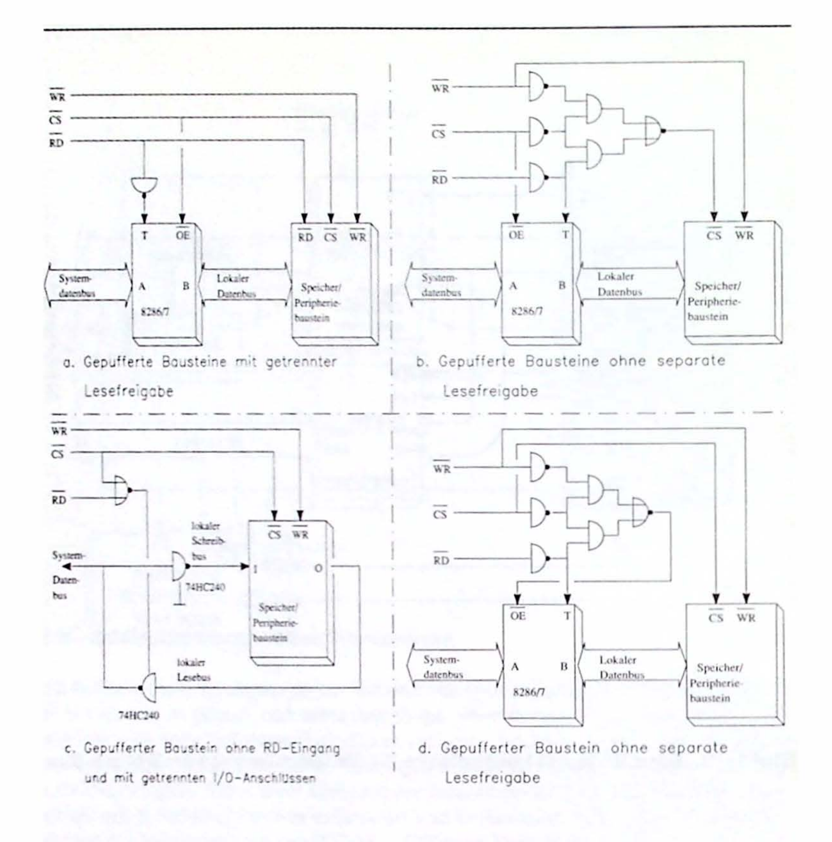
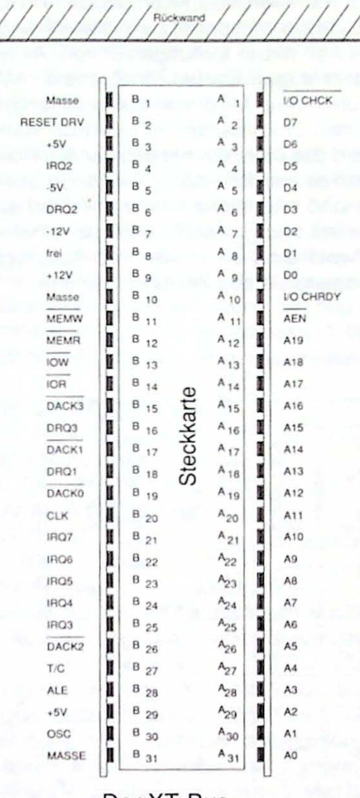
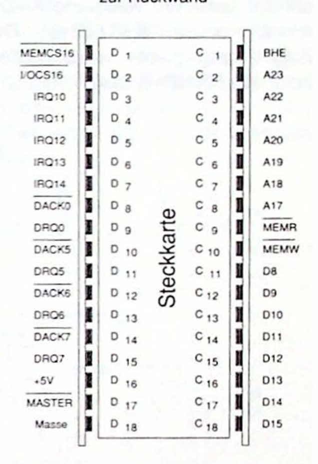


Bild 1-11. Anschlüsse der unterschiedlichen Speicher und Peripheriebausteine an den Datenbus.

Als Beispiel für einen Systembus sei der PC-XT- und der PC-AT-Bus in den IBM-kompatiblen Rechnern genannt. Auf ihm finden sich viele Leitungen, die Peripheriebausteine auf der Mutterplatine und Steckkarten ansprechen. Bild 1-12 a zeigt die Belegung der Slot-Plätze für den XT- und AT-Bus, während Bild 1-12 b ein typisches Beispiel für die Konfigurierung eines Steckkartenaufbaus gibt.



Der XT-Bus



Zusätze im AT-Bus

Bild 1-12 a. Der PC-Bus in IBM-kompatiblen Rechnern

Der wichtigste Teil auf der Steckkarte ist die Adreßdekodierung und die Datenpufferung. Die Dekodierung erfolgt mit dem 8-Bit Magnituden-Komparator 74HC688 und dem 3-zu-8-Dekoder 74HC138. Der IBM-Bus verwendet zur Adressierung des I/O-Bereichs nur die Adreßleitungen A_0 bis A_9. Sieben Leitungen davon, A_3 bis A_9 werden den P-Eingängen des Komparators zugeführt, wo ihr Zustand mit den durch die Dip-Schalter eingestellten verglichen wird. Mit dieser Methode ist die Basisadresse der Steckkarte in 8-Byte-Schritten im gesamten I/O-Bereich wählbar. Durch die Leitung Address Enable $\overline{AEN}$ wird der Baustein bei gültiger Adresse selektiert. Nur wenn das Bit-Muster der Adresse mit dem der Q-Eingänge übereinstimmt, geht der Ausgang $\overline{P=Q}$ an Masse und trägt seinen Teil zur Freischaltung des Dekoders 74HC138 bei. Der Dekoder soll aber ausschließlich bei einem I/O-Zugriff freigegeben werden. Zu diesem Zweck sind die I/O-Schreib-/Lesesignale über ein NAND-Gatter 74HC10 an den Eingang G1 des 74HC138 geführt.

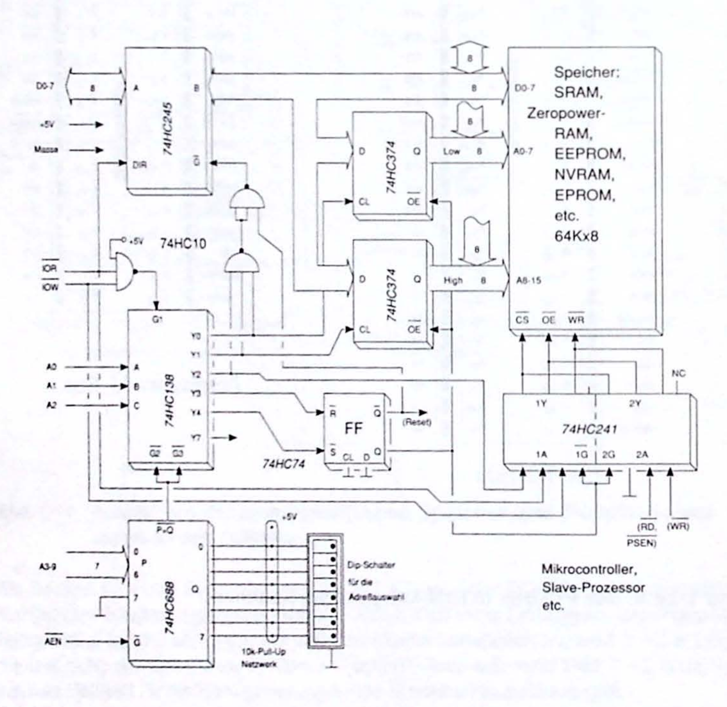


Bild 1-12 b. Typische Konfigurierung einer Steckkarte für den PC-Bus

Der Transceiver 74HC245 trennt einmal den Datenbus von der Steckkarte und verstärkt zum anderen die Signale. Seine Verwendung ist absolut erforderlich, da es sonst zu fehlerhaften Informationen in beide Richtungen kommen kann. Die Richtung des Datenflusses wird durch das Lesesignal bestimmt. In der Regel genügt es, den Chip-Select-Eingang $\overline{G}$ an Masse legen, so daß immer auf die Karte zugegriffen werden kann.

Die restlichen Teile der Karte sind optional. Im vorliegenden Fall ist die Erweiterung so gehalten, daß ein Speicherbaustein oder ein Uhrenchip, z.B. 48T02, beschrieben und gelesen werden kann. Die Adresse für den Speicher muß zunächst als Information über den Datenbus in die Zwischenspeicher 74HC374 fließen, erst dann kann der Speicher über die $\overline{CS}$-Leitung zum Beschreiben oder Lesen freigegeben werden. Um das Low-Byte der Adresse zu setzen, muß ein I/O-Schreibbefehl an die Basisadresse+0 erfolgen; für das Schreiben des High-Bytes ist ein I/O-Schreibbefehl an die Basisadresse+1 notwendig. Der Inhalt der so adressierten Speicherstelle ist über die Basisadresse+2 erreichbar:

```
 MOV DX,Basisadresse+0
 MOV AL,Adresse_Low
 OUT DX,AL
 INC DX
 MOV AL,Adresse_High
 OUT DX,AL
 INC DX
 MOV AL,Wert
 OUT DX,AL    (zum Schreiben in den Speicher)
(IN  AL,DX    zum Lesen des Speichers)
```

In der Regel wird man die Chip-Select-Eingänge $\overline{OE}$ der beiden 74HC374 an Masse legen. In der Schaltung nach Bild 1-12 b sind die Adreßspeicher und der Transceiver durch einen Schreib- oder Lesebefehl an die Basisadresse+4 abschaltbar und durch einen Schreib- oder Lesebefehl an die Basisadresse+3 wieder einschaltbar. An der Basisadresse+3 wird der Ausgang Q des Flip-Flops 74HC74 an Masse gelegt und der $\overline{Q}$-Ausgang an Plus. Durch diese Maßnahme wird ein weiterer Prozessor oder ein Mikrocontroller auf der Karte ein- oder ausgeschaltet. Mit Hilfe des Ausgangs $\overline{Q}$ kann ein Transceiver und weitere Adreßspeicher, die zum Gastprozessor oder Mikrocontroller gehören, freigegeben werden.

Der Speicher ist mit entsprechender Software von Seiten des PCs mit dem Maschinenkode des Mikrocontrollers beschreibbar, der nach seiner Freigabe (Basisadresse+4) das Programm ausführt und die Ergebnisse im Speicher ablegt, wo sie der PC lesen kann. Mit wenigen Handgriffen hat man so eine einfache Möglichkeit zum Entwickeln und Testen von Programmen für andere Prozessoren.

Verzichtet man auf diese Umschaltmöglichkeit, kann die vorliegende Karte auch als EPROM- oder EEPROM-Brenner verwendet werden. Mit den freien Ausgängen des 74HC138 können verschiedene Brennspannungen und verschiedene Speichergrößen selektiert werden. Der Baustein 74HC241, der zwischen den Kontrollsignalen der verschiedenen Systemen umschaltet, ist dann nicht mehr nötig.

1.2 Buskonzepte für Peripheriebausteine

Peripheriebausteine können abhängig von der Art des verwendeten Prozessors entweder im Programm- oder Datenspeicher integriert sein oder einen separaten I/O-Adreßbereich einnehmen. So verfügen die Mikrocontroller über keine eigenen I/O-Befehle, da sie selbst bereits mit I/O-Ports versehen sind. Sie besitzen zwei verschiedene Steuer-Pins, die Lesesignale für den externen Speicher ausgeben. Der Pin Program Store Enable $\overline{PSEN}$ wird bei dem Einlesen eines Befehls aus dem ROM benutzt, der Pin $\overline{RD}$ nur beim Lesen von Daten. Die Mikrocontroller-Familie MCS-51 verfügt über einen 64 K großen Speicherbereich, der nur gelesen werden kann und in dem das Programm und Konstantentabellen untergebracht werden, und über einen 64 K großen Datenspeicherbereich, der beschrieben und gelesen werden kann. Peripheriebausteine können daher nur im zweiten Bereich untergebracht werden. Dadurch, daß die Mikrocontroller bereits I/O-Ports besitzen, kann durch eine spezielle Hardware-Konfiguration dieser Datenbereich um weitere parallele 64-K-Bänke (max. 16 Stück) auf 1 MByte erweitert werden, wovon eine Bank für die Peripheriebausteine reserviert werden kann.

Die Prozessoren der Personal Computer verfügen über eigene I/O-Maschinenbefehle, die auch über die höheren Programmiersprachen zur Konversation mit den Peripheriebausteinen aufgerufen werden können. Bei der Benutzung dieser Befehle schaltet im einfachsten Fall der Pin M/IO den Speicher ab und die Peripherie ein, so daß sich beide denselben Adreß- und Datenbus teilen können. Bei der Verwendung der Statusleitungen $S_{0\text{-}2}$ übernimmt der Buscontroller die Dekodierung und gibt anstelle der CPU den M/IO-Pegel aus. Der Adreßbereich ist in diesem Fall auf 64 K begrenzt. Mit Hilfe der INPUT- und OUTPUT-Befehle können 16-Bit-Daten über das AX-Register oder 8-Bit-Daten über das AL-Register transferiert werden. Die Peripheriebausteine können wie die Speicher auch entweder am gemultiplexten CPU-Bus oder am gepufferten Systembus angeschlossen werden.

In System mit einem 16-Bit-Datenbus können 8-Bit-Peripheriebausteine entweder mit der oberen Hälfte oder der unteren Hälfte des Datenbusses verbunden sein. Eine Gleichverteilung ist zu empfehlen, da dadurch die Busbelastung gleich verteilt wird. Wenn ein Baustein mit der oberen Hälfte des Datenbusses verbunden ist, müssen alle I/O-Adressen ungerade (A_0=1) sein, ist er mit der unteren Hälfte verbunden, müssen sie gerade (A_0=0) sein. Da die Adreßleitung A_0 für einen Zugriff immer Eins oder Null sein wird, kann A_0 nicht als Adreßeingang für die Registerselektierung eines Bausteins verwendet werden. A_0 in Verbindung mit dem Signal $\overline{BHE}$ muß die Chip-Select-Dekoder auswählen, um ein irrtümliches Schreiben an einen anderen zu verhindern. Die gebräuchlichen Techniken, mit deren Hilfe Peripheriebausteine in einem 16-Bit-Datenbus ausgewählt werden, zeigt Bild 1-13.

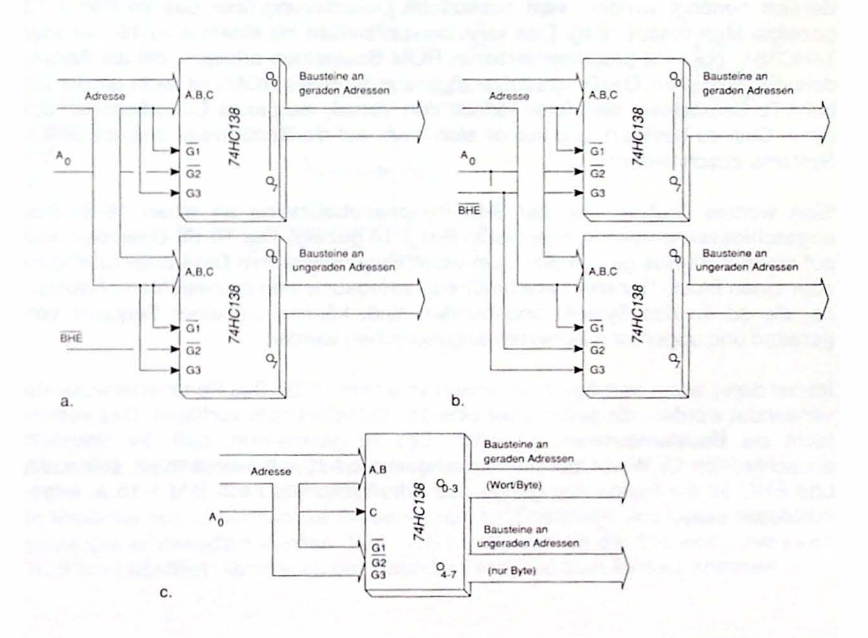


Bild 1-13. Techniken zur Auswahl von Peripheriebausteinen an einen 16-Bit-Bus

Die erste Technik (a) benutzt zwei getrennte 74HC138 , um die Chip-Auswahlsignale für ungerad- und gerad-adressierte Peripheriebausteine. Wenn in diesem Fall ein Wort-Transfer an eine gerade Adresse erfolgt, wird der Baustein mit der folgenden ungeraden Adresse ebenfalls selektiert. Es kann also ein Byte individuell an einen Baustein oder ein Wort (zwei Bytes) an zwei Bausteine gleichzeitig geschrieben werden. Bild 1-13 b beschränkt die Chip-Auswahl auf den Byte-Transfer. Die dritte Technik (Bild 1-13 c) benutzt einfach einen 3-zu-8-Dekoder, um die Auswahl zwischen geraden und ungeraden Adressen zu treffen. Nur wenn ein Wort-Transfer an eine gerade Adresse erfolgt, werden beide Bausteine an der geraden und ungeraden Adresse freigegeben.

Wenn mehr als 256 Bytes an I/O-Bereich oder in den Speicher eingebetteten Bereich benötigt werden, wird zusätzliche Dekodierung über das im Bild 1-13 gezeigte Maß hinaus nötig. Das kann beispielsweise mit einem 4-zu-16-Dekoder 74HC154 oder mit programmierbaren ROM-Bausteinen erfolgen, die als Adreßdekoder fungieren. Die Durchlaufverzögerung durch die ROMs ist leicht größer als bei TTL-Bausteinen; sie bieten jedoch den Vorteil, die ganze Dekodierlogik auf einen Chip zu besitzen, und lassen sich leicht auf die Bedürfnisse des speziellen Systems zuschneidern.

Eine weitere Technik, mit der 8-Bit-Peripheriebausteine an einen 16-Bit-Bus angeschlossen werden können, ist in Bild 1-14 gezeigt. Der 16-Bit-Datenbus wird auf einen 8-Bit-Bus gemultiplext, um einen Byte-orientierten DMA unterzubringen oder einen Block-Transfer zwischen Peripheriebausteinen zu erleichtern. Bausteine, die so an das System angebunden sind, können mit einer Sequenz von geraden und ungeraden Adressen angesprochen werden.

Immer dann, wenn es möglich ist, sollten an einem 16-Bit-Bus Peripheriebausteine verwendet werden, die selbst über eine 16-Bit-Datenbreite verfügen. Das vereinfacht die Bausteinauswahl erheblich. Um zu garantieren, daß der Baustein ausschließlich für Wort-Operationen mit gerader Adresse selektiert ist, sollten A_0 und $\overline{BHE}$ an die Freigabeeingänge des Adreßdekoders nach Bild 1-15 a. angeschlossen sein.

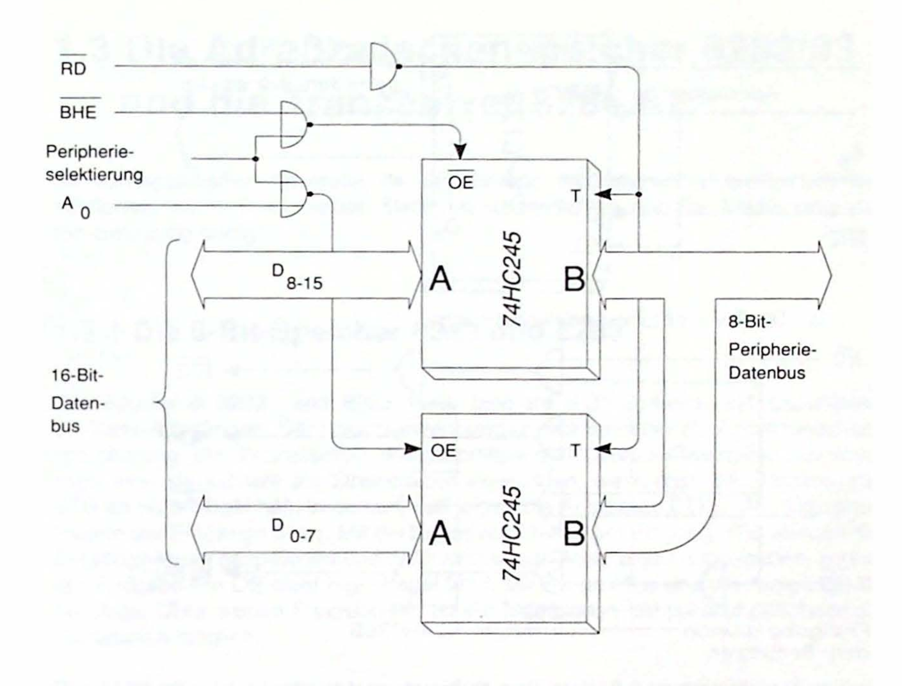


Bild 1-14. 16-Bit- zu 8-Bit-Busübersetzung

In ähnlicher Weise können die Steuersignale für Schreiben und Lesen durch einfache Gatter dekodiert werden (Bild 1-15 b). Werden für die Steuersignale keine Tri-States benötigt, kann auch der Dekoder 74HC138 zum Einsatz kommen.

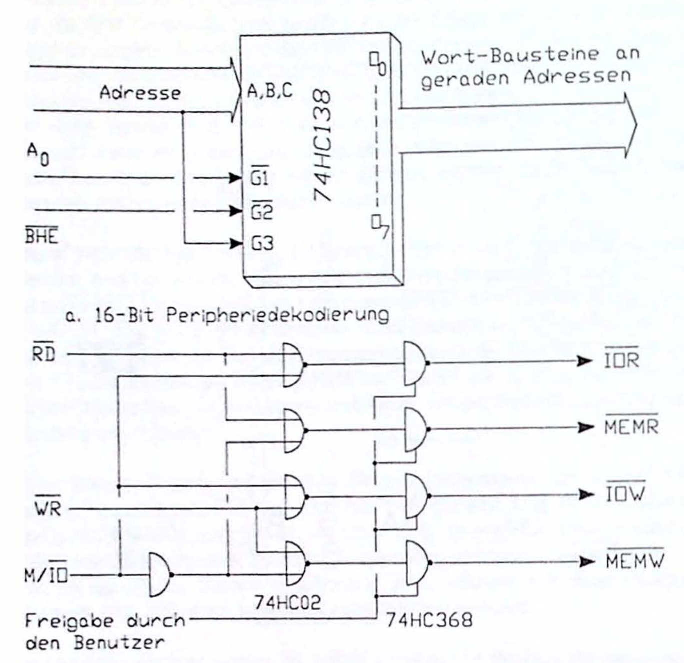


Bild 1-15. 16-Bit-Adreßdekodierung und Dekodierung von Schreib-/Lesesignalen

1.3 Die Adreßzwischenspeicher 8282/83 und die Transceiver 8286/87

Im vorangehenden Abschnitt ist der Einsatz der Bausteine ausführlich beschrieben, so daß an dieser Stelle im wesentlichen nur die Vorstellung der Pin-Belegung erfolgt.

1.3.1 Die 8-Bit-Speicher 8282 und 8283

Die Bausteine 8282 und 8283 sind bipolare 8-Bit-Speicher mit gepufferten Tri-State-Ausgängen. Der Hauptverwendungszweck ist sicher die Adreßzwischenspeicherung bei Prozessoren mit gemultiplextem Adreß-/Datenbus. Daneben kann man sie schlicht als Stromtreiber verwenden, wenn man den Takteingang STB an High-Pegel hält. In diesem Fall folgen die Ausgänge DO_{0-7} den logischen Pegeln der Eingänge DI_{0-7}. Mit der negativen Flanke am Eingang STB werden die Eingänge intern gespeichert und der Inhalt an den Ausgängen ausgegeben, sofern der Freigabe-Pin $\overline{OE}$ nicht High-Pegel führt. Mit $\overline{OE}$ an Plus sind die Ausgänge im Tri-State. Über diesen Freigabe-Pin ist ein Multiplexen bei parallel geschalteten Bausteinen möglich.

Der 8283 gibt die Eingabedaten invertiert aus, so daß beispielsweise aus dem Wert 00 der Wert FF wird. Wenn diese Invertierung konsequent beibehalten wird, ist das kein Nachteil für das System. Diese Tatsache muß nur dann Beachtung finden, wenn Zusatzkarten angeschlossen werden sollen, eine Adreßdekodierung erfolgt, ROMs oder EPROMs ausgelesen werden etc.

Anstelle dieser Zwischenspeicher werden häufig die TTL-Bausteine 74HC373 bzw. 74HC533 (invertierend) benutzt, die zwar preiswerter und funktionsgleich, jedoch nicht pinkompatibel sind.

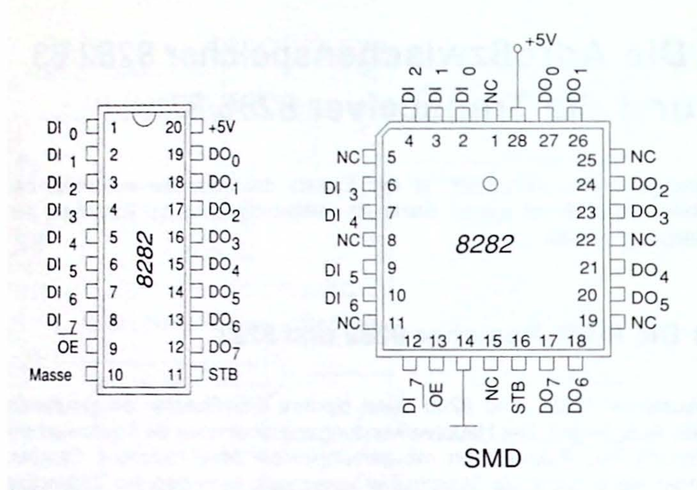


Bild 1-16. Pin-Belegung des 8282

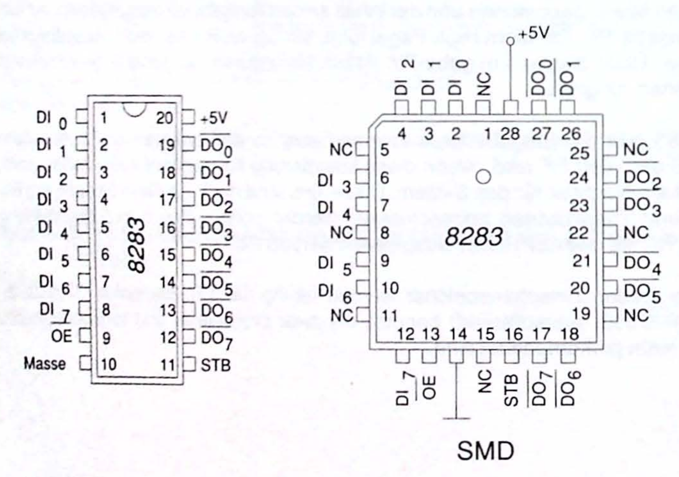


Bild 1-17. Pin-Belegung des 8283

1.3.2 Die 8-Bit-Transceiver 8286 und 8287

Die Bausteine 8286 und 8287 sind bipolare Transceiver mit Tri-State-Ausgängen. Der Baustein 8287 invertiert die Pegel der Dateneingänge an seinen Ausgängen.

Mit dem Eingang T wird die Datenrichtung gesteuert. Führt dieser Pin High-Pegel, sind die Pins $A_{0\text{-}7}$ Eingänge, die Pins $B_{0\text{-}7}$ Ausgänge und die Daten fließen von A nach B. Führt dieser Pin Low-Pegel, sind die Pins $B_{0\text{-}7}$ Eingänge, die Pins $A_{0\text{-}7}$ Ausgänge und die Daten fließen von B nach A. Liegt der Freigabeeingang $\overline{OE}$ an Plus, sind sowohl A als auch B im Tri-State und der Baustein ist abgeschaltet. Da die Anschlüsse A und B sich in ihrer Treibereigenschaft unterscheiden, muß A mit dem lokalen Prozessorbus und B mit dem System-Datenbus verbunden sein. Die Ausgänge A liefern maximal 16 mA, die Ausgänge B maximal 32 mA pro Pin.

Anstelle dieser Transceiver werden häufig die TTL-Bausteine 74HC245 bzw.74HC640 (invertierend) benutzt, die zwar preiswerter und funktionsgleich, jedoch nicht pinkompatibel sind.

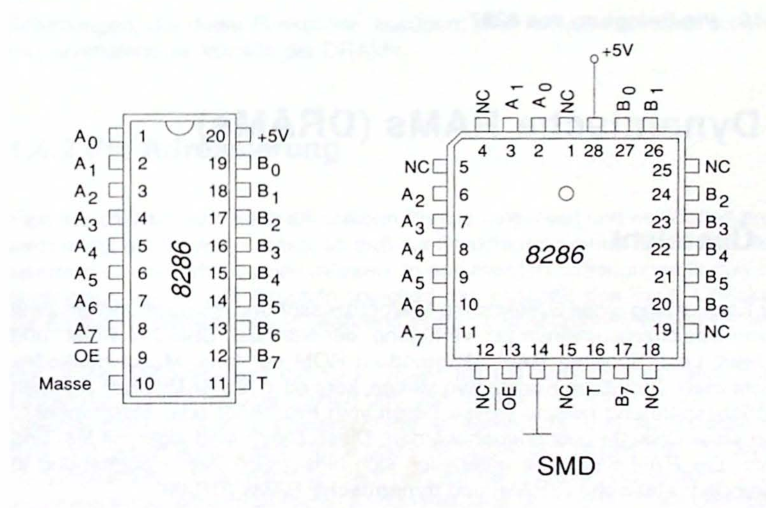


Bild 1-18. Pin-Belegung des 8286

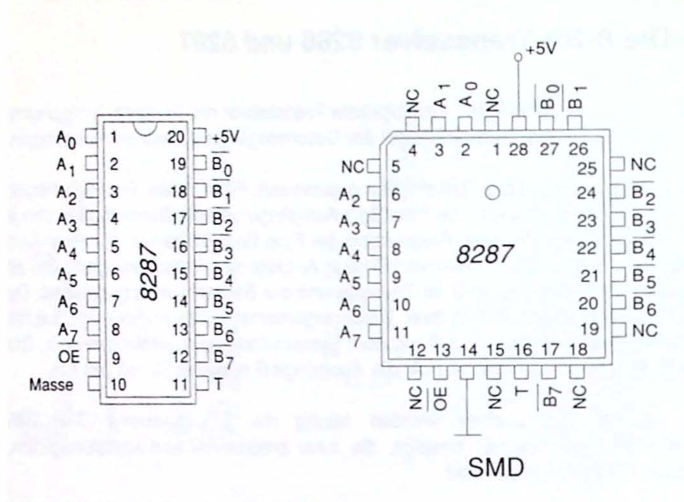


Bild 1-19. Pin-Belegung des 8287

1.4 Dynamische RAMs (DRAMs)

1.4.1 Übersicht

Bei der Entwicklung eines Systems mit Mikroprozessor und Speicher stehen zwei Arten von Speicherbausteinen zur Verfügung: der Nur-Lese-Speicher ROM und der Schreib-Lese-Speicher RAM. Während im ROM nur feste Maschinenkodes oder nicht mehr änderbare Konstanten stehen, können in die RAMs nach Anlegen der Betriebsspannung beispielsweise Daten vom Prozessor oder Maschinenkodes von einer Diskette geschrieben werden. Diese Daten sind jederzeit les- und änderbar. Die RAM-Bausteine unterteilen sich hinsichtlich ihrer Handhabung in zwei Gruppen: **statische** (SRAM) und **dynamische** RAMs (DRAM).

Ein statisches RAM (im folgenden mit SRAM bezeichnet) behält die Daten solange, wie die Betriebsspannung anliegt. Ein dynamisches RAM (im folgenden mit DRAM bezeichnet) verliert auch bei vorhandener Betriebsspannung nach kurzer Zeit (i.a. ≤ 2 ms) seine Information, wenn sein Inhalt nicht zuvor aufgefrischt wurde. Der Auffrischvorgang wird im folgenden mit **Refresh** bezeichnet. Statische

RAMs (SRAM) sind wegen des Wegfalls einer Refresh-Logik leicht zu benutzen, wegen der Verwendung interner Flip-Flops schnell, sind aber etwas teurer und benötigen wegen des fehlenden Adreßmultiplexens mehr Aufwand im Platinenlayout. Dynamische RAMs (DRAM) verfügen heute nur noch über die doppelte Integrationsdichte, sind in kleineren Gehäusen auf dem Markt, ziehen weniger Strom und sind preiswerter pro Bit. Obwohl in den letzten Jahren durch verbesserte Technologie der Integrationsgrad der SRAMs wesentlich zu steigern war und durch hohe Auflagen die Preisdifferenz stark gesunken ist, spielen die DRAMs nach wie vor eine führende Rolle.

Für eine geordnete Funktion benötigen DRAMs eine komplexe Versorgungslogik. Sie schließt ein:

- Adreßmultiplexen
- Richtige Zeitkorrelation zwischen Adreß- und Kontrollsignalen
- Refresh, um den Speicherinhalt zu sichern
- eine Verwaltung, die Konflikte zwischen Refresh- und Schreib-/Lesezyklen verhindert.

Schaltungen, die diese Funktionen ausüben, sind komplex, beanspruchen Platz und schmälern die Vorteile der DRAMs.

1.4.2 Die Adressierung

Fast alle DRAMs sind nach demselben Prinzip aufgebaut und beinhalten pro Baustein meist ein Bit eines Bytes, so daß zur Speicherung eines Bytes acht Bausteine parallel verwendet werden müssen. In den meisten Schaltungen finden sich jedoch neun. Der neunte Baustein speichert das Ergebnis des Parity-Checks , das zur Einfach-Fehlererkennung während eines Speicherzugriffs verwendet wird.

Man kann beispielsweise ein 64-K-DRAM als ein zweidimensionales Feld von Einzelbitspeichern mit 256 Reihen und 256 Spalten betrachten (Bild 1-20). Zwei Dekoder entschlüsseln das binärkodierte Adressen-Low- und das Adressen-High-Byte und geben jeweils eine Reihe und eine Spalte des Feldes frei. Der Inhalt der so ausgewählten Speicherzelle liegt nun auf der internen Leseleitung und kann mit einem entsprechenden Signal an den Ausgang DOUT geführt werden. Ein Baustein, der nach diesem Prinzip funktionierte, benötigte 16 Adreß, zwei Steuer-, zwei Daten- und eine Chip-Select-Leitung. Zusammen mit der Stromversorgung hätte der Chip 23 Anschlüsse. Ein 64-K-DRAM besitzt aber nur 15, also genau acht weniger. Diese Pin-Reduzierung ist nur deswegen möglich, weil dem Baustein über eine 8-Bit-Adreßleitung zuerst die Reihenadresse zugeführt wird, die

der Reihendekoder speichert, danach die Spaltenadresse, die der Spaltendekoder speichert. Dieses Verfahren wird Adreßmultiplexen genannt. Für das Einschreiben der Adresse in die beiden Dekoder werden Steuersignale und damit zwei Pins benötigt. Das erste wird mit RAS (Row Address Strobe), das zweite mit CAS (Column Address Strobe) bezeichnet. Zum Ausgleich dafür verzichten die Bausteine auf einen Lese- ($\overline{RD}$) und einen Chip-Select-Anschluß ($\overline{CS}$) (Bild 1-21).

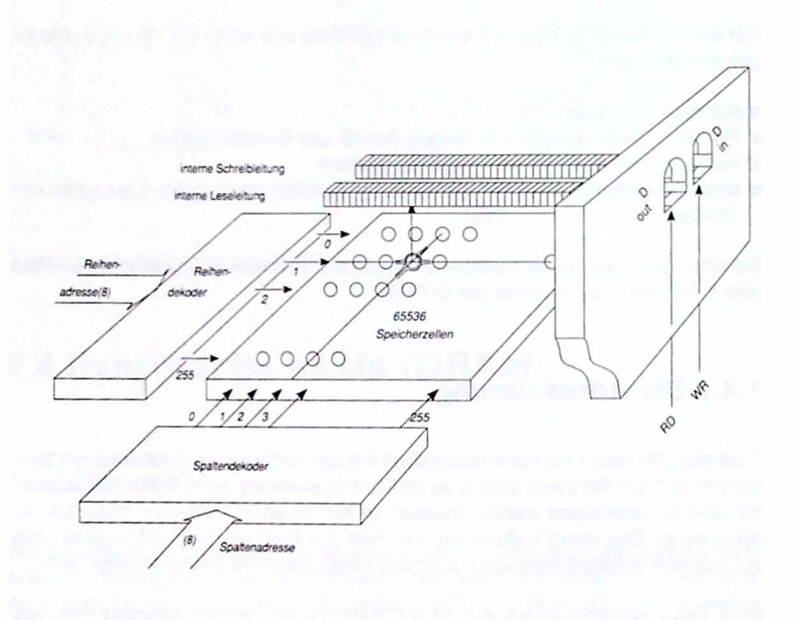


Bild 1-20. Virtueller DRAM-Baustein

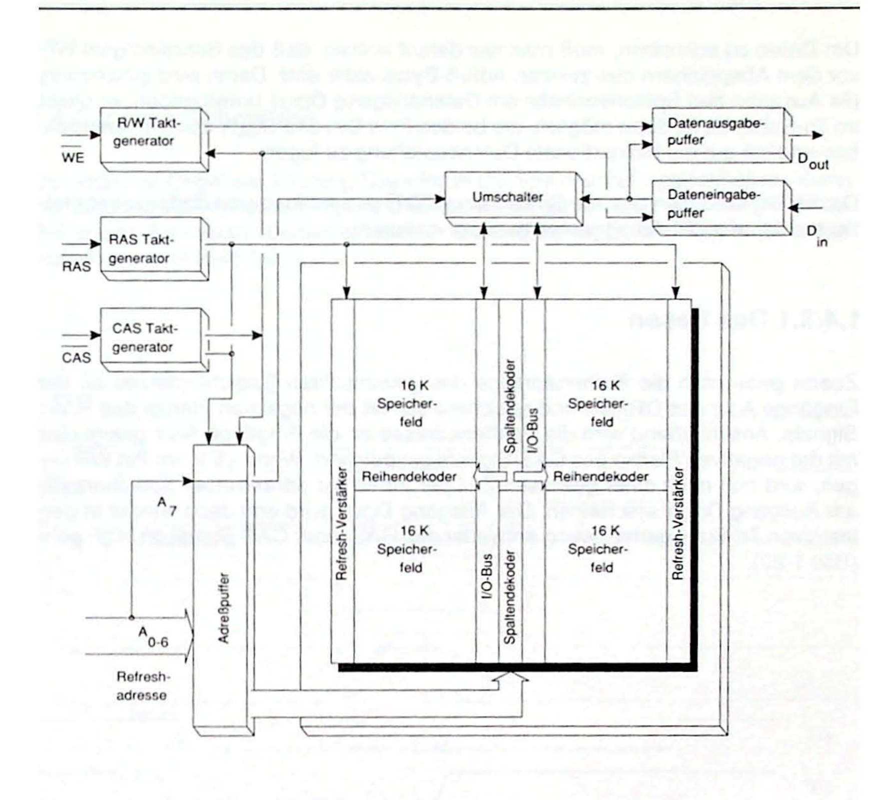


Bild 1-21. Blockdiagramm eines 64-K-DRAM-Bausteins

1.4.3 Die Speicheroperationen

Wie teilt man nun dem Baustein mit, daß Daten geschrieben oder gelesen werden sollen?

Einen Lesezugriff teilt man ihm gar nicht mit, denn sobald sich die Adresse vollständig in den beiden Dekodern befindet, liegt der Inhalt der Speicherzelle am Datenausgang DOUT an. Der Datenausgang DOUT ist ein TTL-kompatibler Tri-State-Ausgang mit einem Fan-Out von zwei TTL-Lasten.

Um Daten zu schreiben, muß man nur darauf achten, daß das Schreibsignal $\overline{WR}$ vor dem Abspeichern des zweiten Adreß-Bytes aktiv wird. Damit wird gleichzeitig die Ausgabe des Speicherinhalts am Datenausgang D_{OUT} unterbunden, er bleibt im Tri-State. Es ist dann möglich, die beiden Pins D_{IN} und D_{OUT} zusammenzufassen und sie auf die bidirektionale Datenbusleitung zu legen.

Da die Signalformen bei den Speicherzugriffen oft sehr kurz und dadurch sehr kritisch sind, wird die Arbeitsweise genauer erläutert:

1.4.3.1 Das Lesen

Zuerst gebe man die Reihenadresse des gewünschten Speicherplatzes an die Eingänge $A_{0\text{-}7}$ des DRAMs und speichere sie mit der negativen Flanke des $\overline{RAS}$-Signals. Anschließend wird die Spaltenadresse an die Eingänge $A_{0\text{-}7}$ gelegt und mit der negativen Flanke des $\overline{CAS}$-Signals gespeichert. Wenn +5 V am Pin $\overline{WR}$ liegen, wird nun nach einer gewissen Zeit der Inhalt der adressierten Speicherzelle am Ausgang D_{OUT} erscheinen. Der Ausgang D_{OUT} wird erst dann wieder in den inaktiven Tri-State gehen, wenn entweder das $\overline{RAS}$- oder $\overline{CAS}$-Signal an High geht (Bild 1-22).

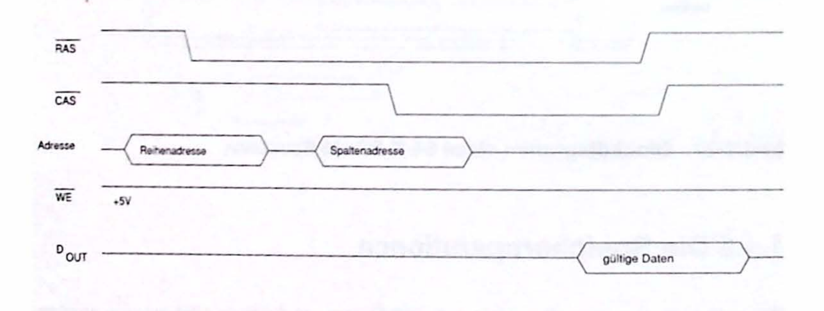


Bild 1-22. Signalformen beim Lesen

1.4.3.2 Das Schreiben

Der logische Pegel am Eingang D_{IN} wird in die Speicherzelle geschrieben, wenn der Eingang $\overline{WR}$ vor dem Eingang $\overline{CAS}$ an Masse geht. D_{OUT} bleibt im Tri-State. Bei dieser Schreibart können die beiden Daten-Pins D_{OUT} und D_{IN} miteinander verbunden sein (Bild 1-23).

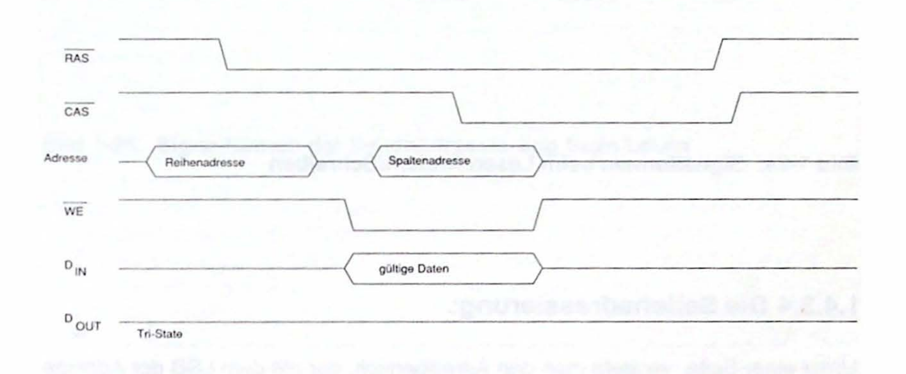


Bild 1-23. Signalformen beim Schreiben

1.4.3.3 Lesen/Ändern/Schreiben

Trift das $\overline{WR}$-Signal nach dem $\overline{CAS}$-Signal ein, erscheint zunächst der Inhalt der Speicherzelle am Ausgang D_{OUT}. Nachfolgend wird mit dem $\overline{WR}$-Signal der logische Pegel am Eingang D_{IN} in die Speicherzelle übernommen. Um diesen Vorgang durchzuführen, müssen die beiden Daten-Pins mit getrennten Datenbussen verbunden sein. Sind in diesem Falle die beiden Daten-Pins miteinander verbunden, ist eine Änderung des Inhalts nicht möglich, da D_{OUT} über D_{IN} in den Baustein zurückgeschrieben wird. Der Modus unterscheidet sich in diesem Fall nicht vom gewöhnlichen Schreibzyklus (Bild 1-24).

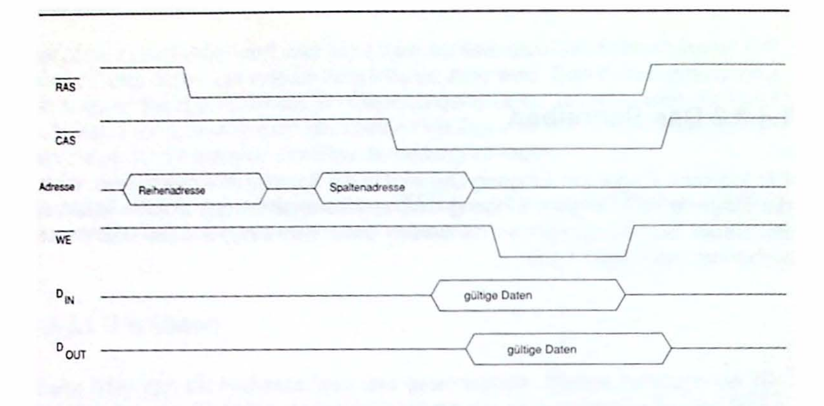


Bild 1-24. Signalformen beim Lesen/Ändern/Schreiben

1.4.3.4 Die Seitenadressierung:

Unter einer Seite versteht man den Adreßbereich, der mit dem LSB der Adresse (acht Bits) erreicht werden kann. Eine Seite umfaßt somit 256 verschiedene Adressen; der Adreßbereich von 64 K besteht aus 256 Seiten, da 256•256 = 64 K.

Die Seitenadresse (MSB) wird dem Reihendekoder zugeführt und in ihm mit dem RAS-Signal gespeichert. Nachfolgend ändern sich nur noch die Spaltenadressen, die mit dem CAS-Signal übernommen werden. Ein RAS-Signal wird erst wieder ausgegeben, wenn eine neue Seite adressiert werden soll. Diese Methode gestattet ein Lesen oder Schreiben mit hoher Geschwindigkeit, wie es bei einem Datentransfer mit einem DMA-Kontroller wünschenswert ist (Bild 1-25 und 1-26).

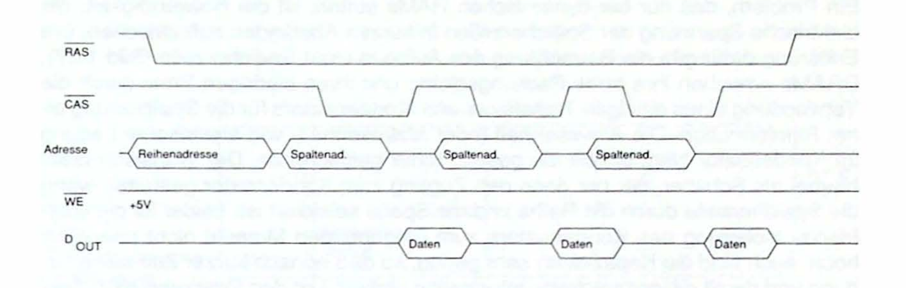


Bild 1-25. Signalformen der Seitenadressierung beim Lesen

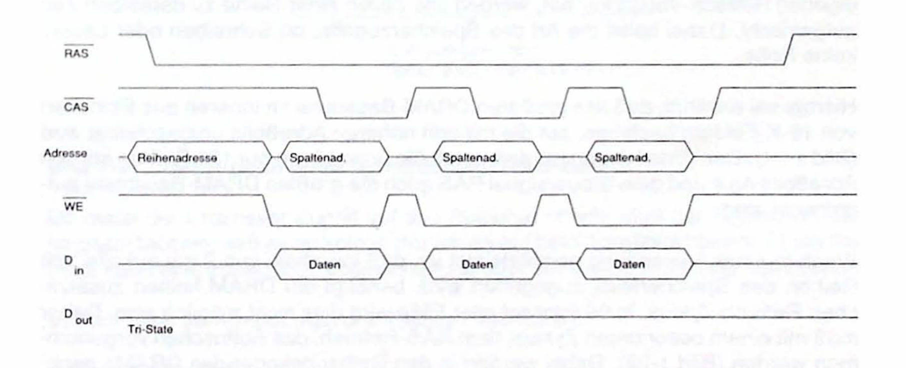


Bild 1-26. Signalformen der Seitenadressierung beim Schreiben

1.4.4 Der Refresh

Ein Problem, das nur bei dynamischen RAMs auftritt, ist die Notwendigkeit, die elektrische Spannung der Speicherzellen in kurzen Abständen aufzufrischen. Die Erklärung dafür gibt die Betrachtung des Aufbaus einer Speicherzelle (Bild 1-27). DRAMs erreichen ihre hohe Packungsdichte und ihren niedrigen Preis durch die Verwendung eines einzigen Transistors und Kondensators für die Speicherung einer Bitinformation. Die Anwesenheit (oder Abwesenheit) von elektrischer Ladung im Kondensator zeigt ein Bit als gesetzt (oder gelöscht) an. Der Transistor dient hierbei als Schalter, der nur dann den Zugang zum Kondensator gestattet, wenn die Speicherzelle durch die Reihe und die Spalte selektiert ist. Leider ist die elektrische Isolierung des Kondensators zum umgebenden Material nicht unendlich hoch, auch sind die Kapazitäten sehr gering, so daß er nach kurzer Zeit seine Ladung und damit die gespeicherte Information verliert. Um den Datenverlust zu vermeiden, muß sein Zustand von Zeit zu Zeit ausgelesen, die Ladung durch eine besondere Schaltung aufgefrischt und der Kondensator mit seinem ursprünglichen Wert zurückgeladen werden. Der dafür notwendige Schaltkreis ist in jedem DRAM integriert und wird Refresh-Verstärker genannt. Die meisten DRAMs müssen ihren Inhalt innerhalb von 2 ms aufgefrischt haben, damit der Inhalt nicht verloren geht. Modernere Bausteine kommen mit einem Intervall von 4 ms aus. Ein Auffrischen einer Speicherzelle erfolgt genau dann, wenn eine Zelle durch eine dekodierte Reihe ausgewählt ist. Die Selektierung durch den Spaltendekoder ist dabei nicht nötig (Bild 1-20). Da eine Reihe immer 128 Zellen umfaßt und da jede Spalte ihren eigenen Refresh-Verstärker hat, werden alle Zellen einer Reihe zu derselben Zeit aufgefrischt. Dabei spielt die Art des Speicherzugriffs, ob Schreiben oder Lesen, keine Rolle.

Hierbei sei erwähnt, daß alle größeren DRAM-Bausteine im Inneren aus Einheiten von 16-K-Feldern bestehen, auf die mit den höheren Adreßbits umgeschaltet wird (Bild 1-21). Der Vorteil davon ist, daß durch die Anwahl von nur 128 Reihen mit den Adreßbits A_{0-6} und dem Steuersignal $\overline{RAS}$ auch die größten DRAM-Bausteine aufgefrischt sind.

Wenn in einer Anwendung gewährleistet ist, daß innerhalb von 2 ms auf alle 128 Reihen des Speicherfelds zugegriffen wird, benötigt ein DRAM keinen zusätzlichen Refresh-Zyklus. In 99 Prozent aller Fälle wird dies nicht möglich sein. Daher muß mit einem besonderen Zyklus, dem $\overline{RAS}$-Refresh, das Auffrischen vorgenommen werden (Bild 1-28). Dabei werden in den Reihendekoder des DRAMs nacheinander 128 Adressen, von 00 bis 7F, mittels $\overline{RAS}$-Signal getaktet. Da das $\overline{CAS}$-Signal nicht benötigt wird, werden weder Daten geschrieben noch gelesen.

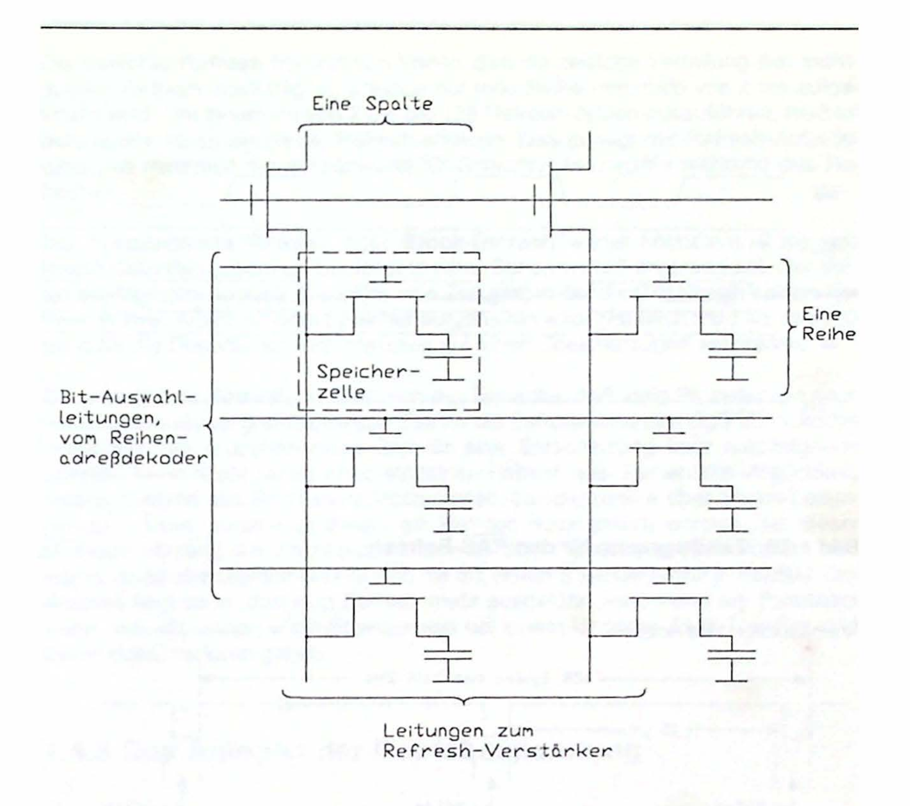


Bild 1-27. Der Aufbau einer dynamischen RAM-Zelle

Da meist der Prozessor Zugriff auf den Speicher nimmt, muß der Refresh-Zyklus so gestaltet sein, daß es zu keinen Konflikten auf dem Adreßbus kommt. Einen Refresh kann eine zusätzliche Logikschaltung oder ein DRAM-Kontroller vornehmen.

In der Praxis verwendet man drei Refresh-Techniken:

- der verteilte Refresh
- der konzentrierte oder Block-Refresh
- der versteckte Refresh

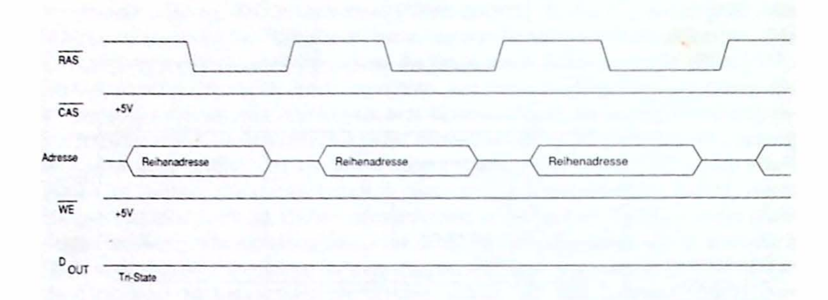
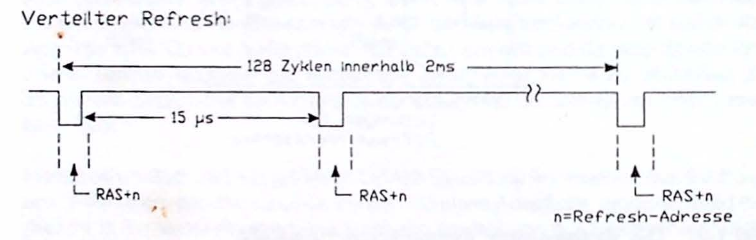


Bild 1-28. Zeitdiagramm für den $\overline{RAS}$-Refresh



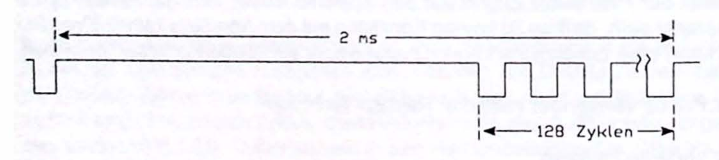


Bild 1-29. Zeitdiagramm der Refresh-Zyklen

Der **verteilte Refresh** benutzt den Vorteil, daß die zeitliche Verteilung des individuellen Refresh unwichtig ist, solange nur jede Reihe innerhalb von 2 ms aufgefrischt wird. Um innerhalb von 2 ms die 128 Refresh-Zyklen auszuführen, muß im Schnitt alle 15 µs ein Einzel-Refresh erfolgen. Das genügt der Refresh-Anforderung und minimiert die Verzögerung für Schreib-/Lesezugriffe während des Refreshs.

Der **konzentrierte Refresh** oder **Block-Refresh** wartet höchstens 2 ms und frischt dann den gesamten Speicher in einer Serie von 128 Impulsen auf. Der Vorteil der Methode liegt darin, daß es eine Zeit gibt, in der die CPU durch keinen Refresh in ihrer Arbeit mit dem Speicher aufgehalten wird. Natürlich muß sie nachfolgend für die Periode von 128 Impulsen auf einen Speicherzugriff verzichten.

Der **versteckte Refresh** profitiert von der Tatsache, daß viele Prozessoren nach der Befehlseinlesung eine gewisse Zeit für die Dekodierung des Operationskodes benötigen. Sie brauchen diese Zeit für eine Entscheidung über nachfolgende Schritte. Meist reicht sie für einen einzelnen Refresh aus. Besteht die Möglichkeit, diesen Zustand des Prozessors festzustellen, beispielsweise über Status-Leitungen etc., kann unmittelbar darauf ein Refresh durchgeführt werden. Bei dieser Methode bemerkt der Prozessor nichts von dem Refresh, er bleibt für ihn versteckt. Außerdem kommt ein Refresh nie mit einem Speicherzugriff in Konflikt. Der Nachteil liegt darin, daß kein Refresh mehr ausgeführt wird, wenn der Prozessor seine Aktivität stoppt, wie beispielsweise bei einem längeren DMA-Transfer, und Daten dabei verloren gehen.

1.4.5 Das Anlegen der Betriebsspannung

Wegen der im Chip integrierten Kondensatoren zeigt ein DRAM-Baustein ein anderes Einschaltverhalten als statische Bausteine. Einige Punkte sind im Hardware-Konzept zu beachten:

- Das Ansteigen der positiven Betriebsspannung sollte recht langsam erfolgen. Liegt die Anstiegszeit bei ca. 10 µs, wird der Baustein nicht dynamisch arbeiten und ein hoher Strom durch ihn fließen. Empfohlen ist eine Anstiegszeit, die nicht kürzer als 100 µs ist. Das läßt sich mit einem Kondensator entsprechender Kapazität und vorgeschaltetem Widerstand erreichen.

- Nach der Stabilisierung der positiven Betriebsspannung sollte eine Zeit von mehr als 500 µs verstreichen, in der der Baustein ruhen muß. Danach sind zur Initialisierung der dynamischen Zellen acht oder mehr Refresh-Zyklen auszuführen. Erst danach kann der Baustein vom System benutzt werden.

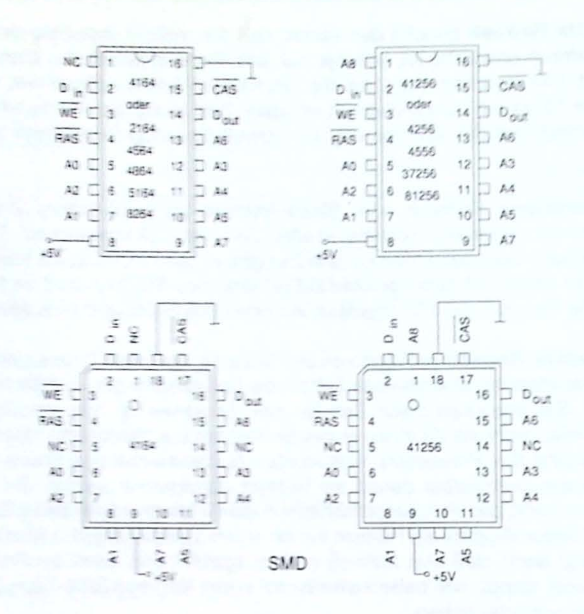


Bild 1-30. Die Pin-Belegung der wichtigsten DRAMs

1.4.6 Anforderungen an das Platinenlayout

Bei der Entwicklung gedruckter Schaltungen ist auf einige Dinge zu achten, wenn man keine unangenehmen Überraschungen erleben will.

Die meisten Systeme für 16-Bit-Prozessoren enthalten 16 bis 32 DRAMs mit einer Speicherkapazität von 1 MByte. Die Ausgänge der Adreßtreiber gehen zu jedem DRAM-Baustein. Ähnlich viele Eingänge haben die $\overline{\text{RAS}}$- und $\overline{\text{CAS}}$-Steuersignale zu treiben. Die Realisierung eines solchen Speichersystems ist nicht besonders problematisch, wenn man die Schwierigkeiten kennt und etwas sorgfältig bei der Planung vorgeht.

Die zwei Punkte, die Datenfehler oder gar einen Datenverlust bewirken können, sind:

1. Positive oder negative Spannungsüberspitzen hervorgerufen durch induktive Verbindungen und hohe kapazitive Lasten.

2. Stromspitzen durch Umschalten kapazitiver Lasten

1.4.6.1 Spannungsüberspitzen

Die Verwendung einer großen Zahl von DRAMs hat eine große kapazitive Last zur Folge, die sich aus den Eingangskapazitäten der einzelnen RAMs und den Leitungskapazitäten zusammensetzt. Letztere verbinden sich mit der Induktivität der Leiterbahnen und erscheinen so als komplexe Impedanz für die Treiber. Sie erzeugen bei jedem Pegelwechsel der Treiber Spannungsüberspitzen. Bild 1-31 zeigt Ersatzschaltungen, die den Sachverhalt erläutern: In Bild a ist der Kondensator, der die Leitungs- und Eingangsimpedanz symbolisieren soll, geladen. Wird der Schalter geschlossen, was einem sehr schnellen Pegelwechsel der Treiber entspricht, fließen auf die zuvor positiv geladene Platte des Kondensators die Elektronen und stoßen die überzähligen auf der gegenüberliegenden Platte ab. Diese können wegen des Widerstandes nicht schnell genug zur Masse fließen und sind am Ende der Leitung nun in der Überzahl vorhanden. Erhöhte negative Ladung stellt aber einen Spannungspegel von unter 0 V dar. Im Falle eines abrupten Pegelwechsels von Minus nach Plus, sind die Verhältnisse ähnlich. Beim Schließen des Schalters fließen die Elektronen ruckartig zum Plus-Pol und lassen die abstoßende Wirkung auf die Elektronen der gegenüberliegenden Platte verschwinden. Die zuvor weggedrängten Elektronen fließen aus dem Hintergrund der Leitung nach und lassen dort kurzfristig einen Pegel zurück, der wesentlich über +5 V liegt.

Das Maß dieser Spannungsspitzen hängt ab von der Geschwindigkeit des Pegelwechsels und der Größe der Kapazität. Das kann schlimmstenfalls eine Zerstörung der DRAMs zur Folge haben oder durch Echos bedingte Prellschläge Fehler im Inhalt der Speicher bewirken.

Um diese Spannungsspitzen zu reduzieren, hilft nur ein serieller Widerstand zwischen Treiber und kapazitiver Last. Seine Aufgabe ist es, die Entladung so zu verlangsamen, daß es zu keinem übermäßigen Spannungstoß kommt. Ist der Wert des Widerstandes zu hoch, kommt es zu einer Überdämpfung und der Pegelwechsel an den Eingängen der Speicher erfolgt zu langsam und zu spät. Bei Steuersignalen wird dadurch die Durchlaufverzögerung zu groß und die Bausteine werden nicht mehr richtig getaktet. Zu kleine Werte des seriellen Widerstandes bewirken keine Abhilfe des Problems. Die Werte der benötigten Widerstände

liegen im Bereich von 33 Ω bis 60 Ω und hängen ab von der speziellen Impedanz der Leitung. In jedem Fall muß der richtige Wert des Widerstandes empirisch, d.h. durch Betrachtung der Signalform in einem Oszillographen ermittelt werden.

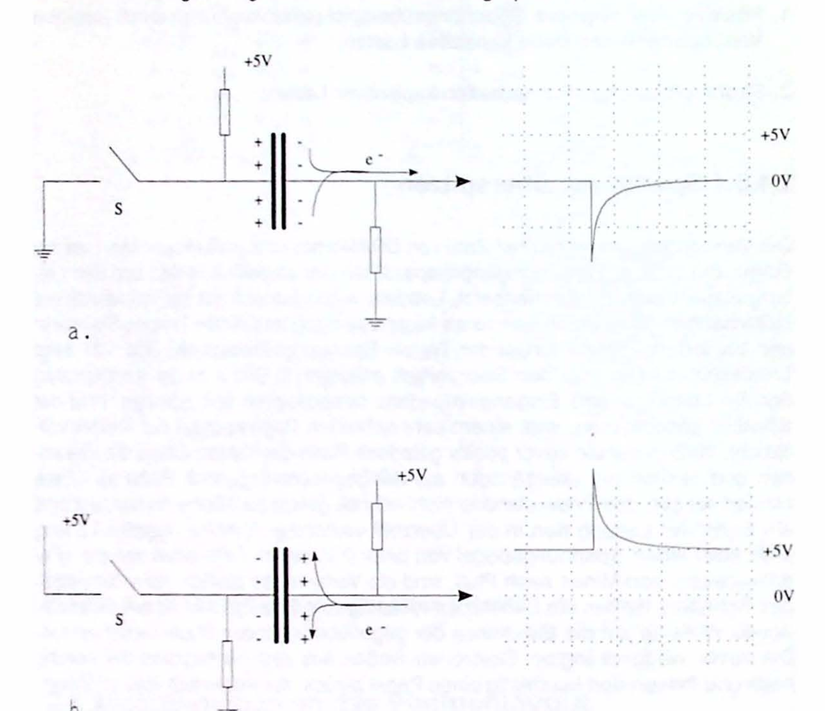


Bild 1-31. Ersatzschaltung zur Entstehung von Spannungsüberspitzen

1.4.6.2 Stromspitzen

Stromspitzen entstehen, wenn zahlreiche Eingangskapazitäten umgeladen werden, d.h. immer bei sehr schnellem Pegelwechsel. Das Resultat ist ein kurzfristig hoher Stromstoß, der wegen des auch noch so geringen Widerstandes der Stromversorgungsleitungen nicht augenblicklich durch das Netzteil ausgeglichen werden kann. Das bewirkt ein übermäßiges Abfallen der Spannung in der Nähe des Chips. Die Folge kann eine unkorrekte Arbeitsweise des Bausteins sein. Um diese Spannungsflaute aufzufangen, ist es notwendig, möglichst nahe an den Stromversorgungsanschlüssen eines jeden Chips einen 0,1 µF keramischen Entkoppelkondensator anzubringen.

Kapitel 2

2 Die Anschlußbelegungen der wichtigsten Mikrocontroller und Prozessoren der 80er Reihe

In diesem Abschnitt werden die wichtigsten Vertreter der 80er Prozessoren mit ihren Pin-Belegungen vorgestellt. Ein kurzer Kommentar konzentriert sich im wesentlichen auf die Erklärung der jeweiligen Busstruktur, so daß die Verwendung der in den weiteren Kapiteln beschriebenen Peripheriebausteine in einem speziellen Prozessorensystem leichter zu verstehen ist. Auch wenn man eine bestehende Schaltung untersuchen möchte, ist die Kenntnis der Pin-Belegung und der Prozessorstruktur immer von Vorteil. Beim Studium der Peripheriefunktionen hat sich gezeigt, daß immer wieder ein Blick auf den betrachteten Prozessor für das Verständnis nötig ist. Daher konzentriert sich die Beschreibung der weiteren Prozessoranschlüsse auf solche Pins, die im direkten Zusammenhang mit der Bussteuerung oder der Kommunikation mit den Peripheriebausteinen stehen.

2.1 Die Mikrocontroller-Familie MCS-48

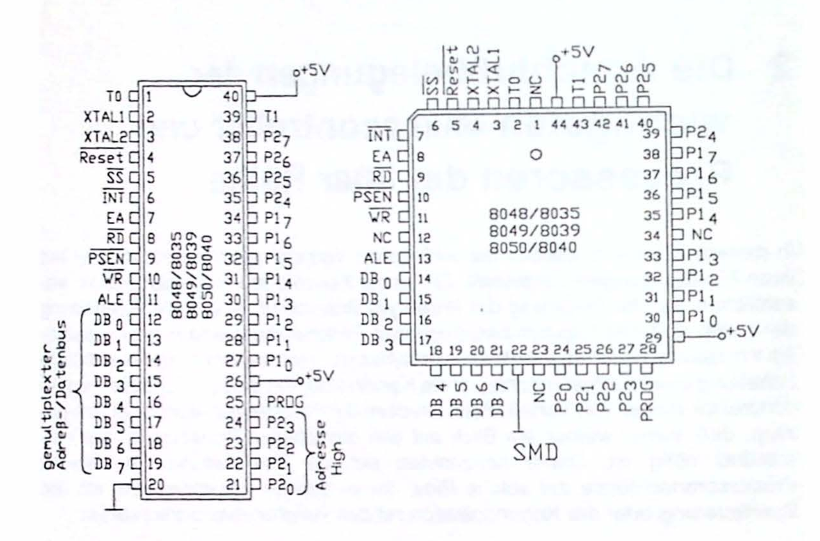


Bild 2-1. Pin-Belegung der MCS-48-Mikrocontroller-Familie

Von den Bausteinen der MCS-48-Mikrocontroller gibt es Vertreter mit integriertem ROM (8048/49/50) von 1 KByte bis 4 KByte Größe und Versionen ohne ROM (8035/39/40). Die ROM-Versionen besitzen einen internen Programmspeicher und führen keinen externen Zugriff aus, wenn die interne Kapazität ausreicht. Beim Überschreiten der Bereichsgrenze wird automatisch ein Zugriff auf einen externen Programmspeicher vorgenommen.

Der Zugriff auf den internen Programmspeicher läßt sich durch High-Pegel am External-Access-Eingang EA auf einen externen Programmspeicher umlenken, sodaß die ROM-Versionen identisch zu den Versionen ohne ROM arbeiten. Bei einem Zugriff auf einen externen Programmspeicher stellen die Pins DB_{0-7} den gemultiplexten Adreß-/Datenbus dar. Die vier höherwertigen Bits der Adresse werden

an den Pins $P2_{0-3}$ ausgegeben. Das Adressen-Low-Byte, das über die Pins DB_{0-7} ausgegeben wird, muß mit dem ALE-Signal in einem externen Zwischenspeicher aufgefangen werden. Es stehen zwölf Adreßleitungen zur Verfügung, womit maximal 4 K adressiert werden können. Auf den Einsatz eines Transceivers als Datenpuffer wird man verzichten können.

Für eine Kommunikation mit den Peripheriebausteinen kennt die MCS-48-Familie vier Befehle. Die Befehle MOVX A,@R und INS BUS aktivieren das $\overline{RD}$-Signal und lesen Daten aus der Peripherie, die Befehle MOV @R,A und OUTL BUS aktivieren das $\overline{WR}$-Signal und schreiben Daten in die Peripherie. Da die Adressen nur über die DB-Pins ausgegeben werden, kann ohne eine Bankumschaltung mittels freier Portpins nur ein Adreßbereich von 256 Bytes angesprochen werden. Das ist ausreichend für den Anschluß einer Vielzahl von Bausteinen.

Das Signal ALE ist auch bei den ROM-Versionen bei jedem Maschinenzyklus aktiv und dient somit neben dem Taktsignal für die Adreßzwischenspeicherung als Taktausgang für Peripheriebausteine, die getaktet werden müssen. Das Signal Program Store Enable $\overline{PSEN}$ wird nur dann aktiv, wenn Befehle oder Konstanten aus einem externen Programmspeicher gelesen werden; es ist also das Lesesignal für den externen Programmspeicher. Die Leitung $\overline{RD}$ wird hierbei nicht aktiv.

Die verbleibenden Pins sind im wesentlichen Portein-/ausgänge sowie Steuerleitungen für den internen Programmablauf. Die Mikrocontroller besitzen zwei Testeingänge T0 und T1, deren logische Pegel per Befehl getestet werden und zu Programmsprüngen führen können. T1 dient ferner als Eingang für eine Zählung externer Ereignisse. Low-Pegel am Eingang $\overline{INT}$ kann nach dessen Freigabe einen Interrupt auslösen oder aber wie T0 und T1 als testbarer Eingang zur Verfügung stehen. Mit Masse am Single-Step-Eingang $\overline{SS}$ wird die Befehlsausführung angehalten. An die beiden Eingänge XTAL werden die externen Komponenten (Quarz und Kondensatoren) des internen Schwingkreises angeschlossen.

Der Porterweiterungsbaustein 8243 wurde speziell für die MCS-48-Familie entwickelt. Seine Ein-/Ausgänge werden mit den Pins $P2_{0-3}$ direkt verbunden. Zum Takten dieses I/O-Expanders dient der Ausgang PROG. Seine Aktivitäten werden durch spezielle Befehle gesteuert.

Im 40poligen Gehäuse gibt es den Pin 26 (Pin 29 SMD), der immer mit +5 V verbunden sein muß, da man über ihn die Daten im internen RAM, das eine Größe von 64 bis 256 Bytes aufweisen kann, auch bei Abwesenheit der Betriebsspannung mit Hilfe einer Batterie die Daten aufrecht erhalten kann. Fällt auch hier die Spannung ab, gehen die Daten verloren.

2.2 Die Mikrocontroller-Familie MCS-51

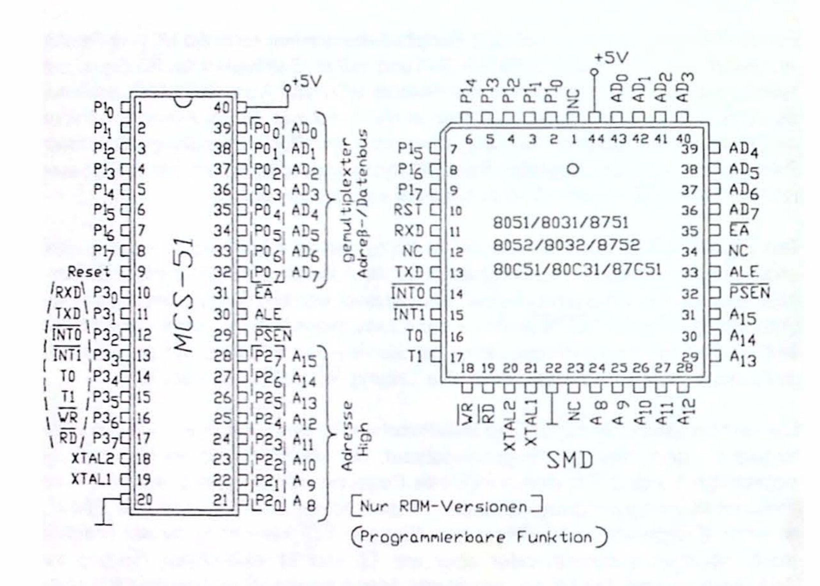


Bild 2-2. Pin-Belegung der MCS-51-Mikrocontroller-Familie

Von den Bausteinen der MCS-51-Mikrocontroller gibt es Vertreter mit integriertem ROM (8051/52 und 8751) von 4 KByte bis 8 KByte Größe und Versionen ohne ROM (8031/32). Die ROM-Versionen besitzen einen internen Programmspeicher und führen keinen externen Zugriff aus, wenn die interne Kapazität ausreicht. Beim Überschreiten der Bereichsgrenze wird automatisch ein Zugriff auf einen externen Programmspeicher vorgenommen. Alle Bausteine besitzen einen 128 bis 256 Byte großen internen RAM-Bereich, in dem häufig benötigte Konstanten oder Zwischenergebnisse abgelegt werden können.

Der Zugriff auf den internen Programmspeicher läßt sich durch Masse am External-Access-Eingang $\overline{EA}$ auf einen externen Programmspeicher umlenken, sodaß die ROM-Versionen identisch zu den Versionen ohne ROM arbeiten. Bei einem Zugriff auf einen externen Programmspeicher stellen die Pins AD_{0-7} den ge-

multiplexten Adreß-/Datenbus dar. Das High-Byte der Adresse wird an den Pins A_{8-15} ausgegeben, die mit den Port-2-Informationen gemultiplext werden. Für das Adressen-High-Byte ist keine Zwischenspeicherung nötig, wohl aber für den Inhalt des Port 2, der in diesem Fall nur als Ausgang dienen kann. Das Adressen-Low-Byte, das über die Pins AD_{0-7} ausgegeben wird, muß mit dem ALE-Signal in einem externen Zwischenspeicher aufgefangen werden. Es stehen 16 Adreßleitungen zur Verfügung, womit maximal 64 KByte adressiert werden können.

Zum Lesen von externen Speichern und Peripheriebausteinen stehen der MCS-51-Familie zwei getrennte Steuerleitungen zur Verfügung, die je nach Art des Befehls aktiviert werden. Bei der Befehlseinlesung wird die Program-Store-Enable-Leitung $\overline{PSEN}$ aktiv und gibt den Programmspeicher frei. Zum Lesen von Daten dient das $\overline{RD}$-Signal. Somit erfolgt automatisch eine physikalische Trennung von 64 KByte Programm- und 64 KByte Datenspeicher. Peripheriebausteine werden in den Datenspeicher integriert und wie RAM-Bausteine adressiert. Da die Bausteine der MCS-51-Familie selbst über eine ausreichende Zahl an I/O-Pins verfügen, kennt ihr Befehlssatz keinen eigenen I/O-Befehl. Es ist auch sehr leicht möglich, mit Hilfe eines freien Portpins dem Peripheriebereich eine eigene Bank zuzuweisen. Das muß natürlich in der Software Berücksichtigung finden. Man kann wahlweise nur einen 256-Byte-Bereich oder den gesamten 64-K-Bereich adressieren. Durch eine portgesteuerte Bankumschaltung ist ein sehr großer Datenbereich von 64 KByte bis theoretisch 4194 MByte verfügbar. Das $\overline{WR}$-Signal wird beim Beschreiben des Datenspeichers und damit der Peripheriebausteine aktiv.

Echte bidirektionale Ports sind nur der Port 1 und mit Einschränkungen der Port 3. Den Pins des Port 3 kann durch die Software alternative Funktion zugewiesen werden, die im Falle ihrer Benutzung die Zahl der Portpins weiter reduzieren. Zum Schreiben oder Lesen aus dem Datenspeicher werden die Pins $P3_{6,7}$ für die entsprechenden Steuersignale benutzt. Die Eingänge T0 und T1 können benutzt werden, um die beiden internen 16-Bit-Zähler zu steuern. Zwei Interrupt-Eingänge $\overline{INT0}$ und $\overline{INT1}$ dienen für eine externe Interruptsteuerung. Schließlich ist mit den Pins RXD und TXD eine serielle Datenübertragung zwischen mehreren Mikrocontrollern oder Peripheriebausteinen möglich.

Die MCS-51-Mikrocontroller-Familie wird ausführlich im Mikrocontroller-Kochbuch behandelt.

2.3 Die Mikrocontroller-Familie MCS-96

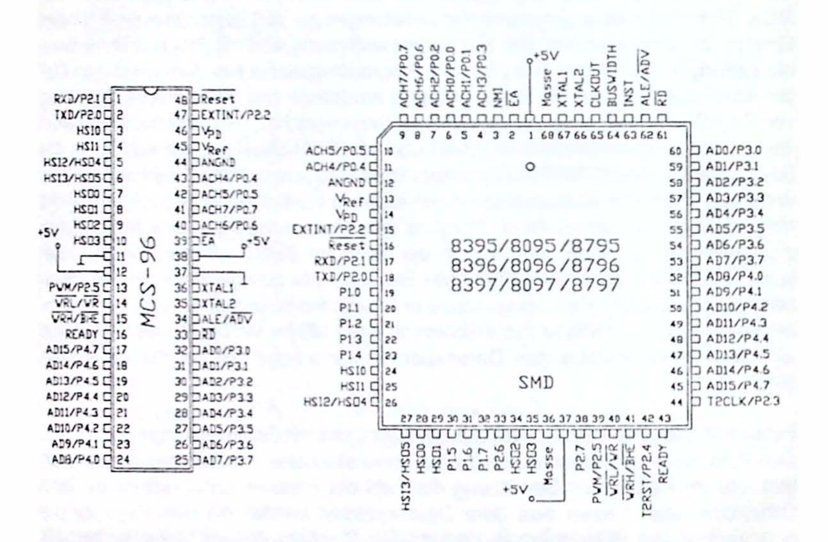


Bild 2-3. Pin-Belegung der MCS-96-Mikrocontroller-Familie

Die Bausteine der MCS-96-Serie sind 16-Bit-Mikrocontroller. Wie alle Mikrocontroller sind sie mit internen 8-K-ROM (839X) oder 8-K-EPROM (879X) erhältlich. Die Versionen ohne ROM tragen die Nummern 809X. Die interne RAM-Größe beträgt bei allen Bausteinen 232 Bytes.

Die Vertreter der MCS-96-Familie besitzen eine 16-Bit-CPU, die sich wesentlich von der herkömmlichen Konzeption unterscheidet. Der ALU angegliedert sind die 232 16-Bit-Register, von denen jedes Akkumulatorfunktion übernehmen kann, so daß die Datenbehandlung durch die ALU nicht mehr den Weg über den Engpaß Akkumulator nehmen muß. Jedes der Register kann Quelle oder Ziel der

ALU-Operation sein, was eine erhebliche Zeitersparnis bringt. Durch besondere Schnelligkeit zeichnen sich auch die anderen integrierten Einheiten aus. Da gibt es eine programmierbare Hochgeschwindigkeits-I/O-Einheit (HSI/O). Sie kann Signale erzeugen oder Ereignisse mit einer Auflösung von 2 µs überwachen. Ferner gibt es einen Analog-zu-Digital-Konverter, einen seriellen Port, einen Pulsbreiten modulierten Ausgang für eine Digital-zu-Analog-Konversation, Watchdog Timer, Zählregister und natürlich I/O-Ports in großer Zahl.

Wird der interne ROM-Bereich überschritten oder liegt der $\overline{EA}$-Pin (External Access) an Masse, holen die Bausteine die Befehle aus einem externen ROM und die Pins AD_{0-15} bilden einen gemultiplexten 16-Bit-Adreß-/Datenbus. Die Bausteine sind so programmierbar, daß die Datenbitbreite auf acht Bit (AD_{0-7}) reduziert wird (ähnlich dem 8088 im Minimum-Modus) oder wahlweise auf einen Bereich zwischen acht und 16 Bit. Im Standard-Modus versorgen die Bausteine einen 16-Bit-Datenbus. Diese Leitungen teilen sich die Pins mit den Ports 3 und 4. Um den Bus zu demultiplexen, wird das ALE-Signal benutzt. Das $\overline{BHE}$-Signal dient in Verbindung mit A_0 einer Dekodierung bei einem Byte- oder Wort-Zugriff auf den Speicher. Dem Lesen von Daten dient das $\overline{RD}$-Signal. Mit dem logischen Pegel am Pin INST kann man zwischen dem Lesen von Daten und Befehlen unterscheiden und somit einen Programm und einen getrennten Datenspeicher unterstützen. Das Schreiben von Daten erfolgt in ähnlicher Weise wie das Lesen mit dem Unterschied, daß nun das $\overline{WR}$-Signal aktiv ist. Beim Schreiben kann zwischen einer externen oder internen Low-High-Byte-Dekodierung wählen. Die externe muß mittels $\overline{BHE}$ und A_0-Signal erfolgen, während die interne beim Schreiben in das Low-Byte das $\overline{WRL}$ bzw. in das High-Byte das $\overline{WRH}$-Signal aktiviert. $\overline{BHE}$ und A_0 werden in diesem Fall nicht benötigt.

Durch eine 16-Bit-Adreßbusbreite kann ein Bereich von 64 K adressiert werden. Die MCS-96-Familie unterscheidet nicht zwischen Programm- und Datenspeicher. Beide liegen in demselben Bereich. Selbst das interne RAM fügt sich in ihn ein. Allerdings erfüllen bedingt durch die Architektur gewisse Adreßbereiche Sonderfunktionen. So kann der Bereich für Peripheriebausteine nur zwischen den Adressen 0100 bis 1FFF und ab 4000 liegen.

2.4 Die Mikroprozessoren 8080/85

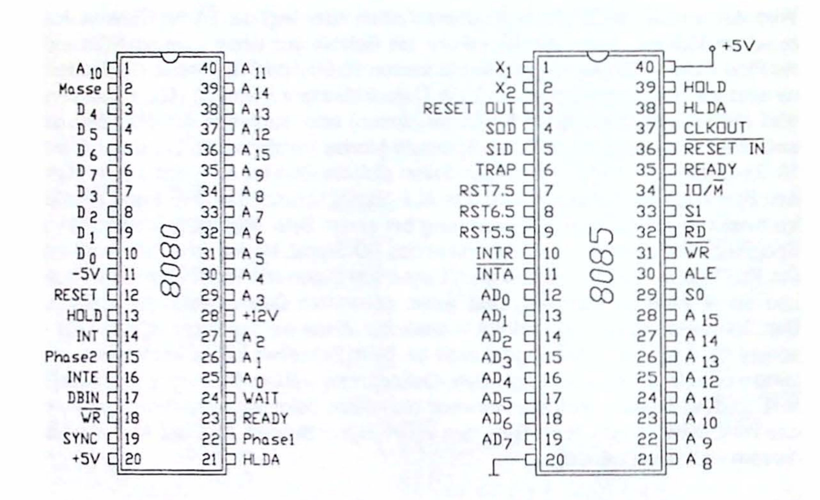


Bild 2-4. Pin-Belegung der Mikroprozessoren 8080/85

Trotz Software-Kompatibilität unterscheidet sich der Hardware-Aufbau der beiden Prozessoren gravierend. Der wesentlichste Unterschied liegt in der Handhabung von Adreß- und Datenbus. Beide Bausteine sind 8-Bit-Prozessoren und besitzen eine Adreßbreite von 16 Bits. Sie können somit einen Bereich von 64 K adressieren. Eine physikalische Trennung von Programm-, Daten- und I/O-Bereich ist nicht vorgesehen. Es gibt einige speziell für das 8080/85-System entwickelte Peripheriebausteine.

Der Baustein 8080 besitzt einen Datenbus, der vom Adreßbus getrennt ist. Somit entfallen eine Zwischenspeicherung der Adresse und die dafür notwendigen Signale. Beim Lesen von Daten gleich welcher Art wird der Ausgang DBIN mit High-Pegel aktiv. Das Signal kann benutzt werden, um Daten aus der Peripherie oder dem Speicher zu lesen. Es entspricht somit einem invertierten RD-Signal.

Das Schreiben von Daten erfolgt in der gewohnten Weise. Adresse und Daten werden ausgegeben, Low-Pegel am Ausgang $\overline{WR}$ schreibt die Daten an die gewünschte Adresse.

Von völlig anderer Seite präsentiert sich der Baustein 8085 . Man erkennt sofort an den Pins AD_{0-7}, daß der Datenbus mit dem Adressen-Low-Byte gemultiplext ist. Zur Adreßzwischenspeicherung ist der Ausgang ALE vorhanden. Außerdem ist mit dem Pin IO/$\overline{M}$ eine Selektierung zwischen Speicher und Peripheriebereich möglich. Weitere Vorteile liegen in der einfachen Spannungsversorgung von +5 V und in der Existenz eines integrierten Oszillators. Mit Hilfe erweiterter Steuersignale ist eine wesentlich komfortablere Kommunikation zwischen Prozessor und Peripheriebausteinen möglich.

2.5 Der Mikroprozessor 8086

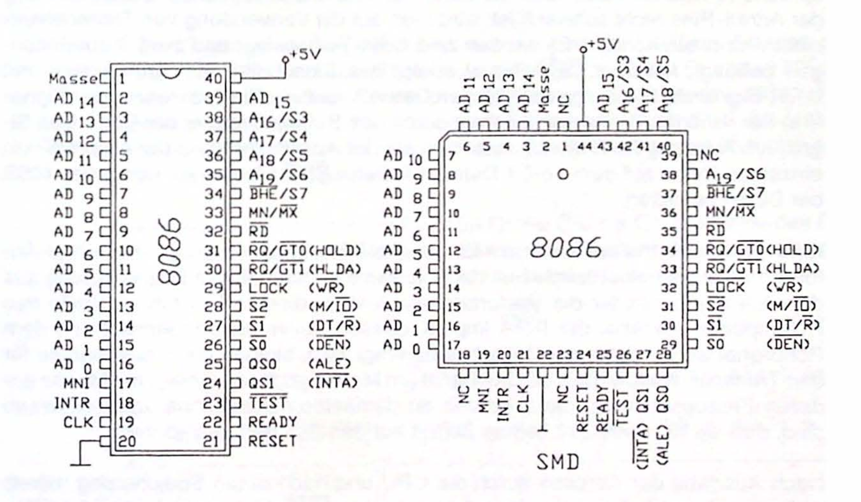


Bild 2-5. Pin-Belegung des Mikroprozessors 8086

Der Baustein 8086 ist ein 16-Bit-Mikroprozessor, der über einen gemultiplexten 16-Bit-Adreß-/Datenbus verfügt. Über die Pins AD_{0-15} fließen sowohl die Daten als auch die unteren zwei Bytes der Adresse. Da die Adreßbits A_{16-19} und das $\overline{BHE}$-Signal mit den Statusleitungen S_{3-7} über dieselben Leitungen gemultiplext ausgegeben werden, müssen nicht nur die Adreßinformationen A_{0-15}, sondern auch die Adreßbits A_{16-19} und das $\overline{BHE}$-Signal mittels ALE-Signal in Zwischenspeichern aufgefangen werden. Es werden also drei 8-Bit-Adreßzwischenspeicher benötigt. Mit den Adreßleitungen ist ein nicht getrennter Programm- und Datenspeicherbereich von einem MByte adressierbar. Erfolgt ein I/O-Befehl, mit dessen Hilfe auf ein Peripheriebaustein zugegriffen wird, sind die Adreßleitungen A_{16-19} immer an Masse, so daß der I/O-Bereich nur 64 K groß sein kann. Der I/O-Bereich ist physikalisch ein vom Daten-/Programmspeicher getrennter Bereich. Zur Unterscheidung beider Bereiche wird das $M/\overline{IO}$-Signal verwendet. Masse an diesem Ausgang selektiert den I/O-Bereich, Plus den Datenspeicher. Der Programmierer braucht sich um diese Steuerung keine Gedanken zu machen, da die Umschaltung automatisch aufgrund des entsprechenden Befehls von der CPU vorgenommen wird.

Nach Ausgabe der Adresse durch die CPU und nach deren Speicherung mittels ALE-Signal, fließen die Daten über die Pins AD_{0-15} und zwar abhängig von der Art des Befehls entweder in (Lesen) oder aus (Schreiben) dem Prozessor. Dementsprechend wird entweder der $\overline{RD}$- oder der $\overline{WR}$-Pin aktiv. Da die Treiberleistung der Adreß-Pins recht schwach ist, wird man auf die Verwendung von Transceivern nicht verzichten können. Es werden zwei 8-Bit-Transceiver und zwei Steuerleitungen benötigt. Mit dem $\overline{DEN}$-Signal erfolgt das Einschalten der Transceiver, mit $DT/\overline{R}$-Signal die Richtungswahl für den Datendurchfluß. Die Aktivierung der Signale in der richtigen Reihenfolge erfolgt durch den Befehlsdecoder der CPU. Das Signal am Ausgang $\overline{BHE}$ dient zusammen mit der Adreßleitung A_0 der Auswahl von einzelnen Bytes auf dem 16-Bit-Datenbus. Führt $\overline{BHE}$ Masse, ist immer das MSB der Daten selektiert.

Eine Ausnahme tritt auf, wenn am Eingang INTR mit High-Pegel eine Interrupt-Anforderung angemeldet wurde und der Prozessor in der Interrupt-Beantwortung aus dem Interruptcontroller die Vektoradresse liest. In diesem Fall tritt an Stelle des $\overline{RD}$-Impulses zweimal der $\overline{INTA}$-Impuls, dessen Zeitverhalten identisch mit dem $\overline{RD}$-Signal ist. Da dafür der Datenbus benötigt wird, bleiben die Steuersignale für den Transceiver aktiv. Das $\overline{LOCK}$-Signal (im Maximum Modus) zeigt mit Masse anderen Prozessoren an, die eventuell an demselben Systembus angeschlossen sind, daß sie für diese Zeit keinen Zugriff auf den Bus nehmen können.

Nach Ausgabe der Adresse durch die CPU und nach deren Speicherung mittels ALE-Signal, erscheint an den Pins A_{16-19} und an $\overline{BHE}$ der interne Status der CPU. S_3 und S_4 zeigen die Herkunft der ausgegebenen Adresse an (Extra Segment ES, Stack Segment SS, Code Segment CS oder Daten Segment DS), womit eine

Auffächerung in vier Bänke möglich ist. S_5 reflektiert den Zustand des Interrupt-Freigabe-Bit im Programmstatuswort, S_6 ist Null und S_7 ist reserviert für künftige Erweiterungen.

Mit High-Pegel am Eingang HOLD können die Aktivitäten der CPU angehalten werden. Sie zeigt ihren inaktiven Zustand durch High am Ausgang HLDA an.

Mit dem Befehl "WAIT" wird der Zustand des Eingangs $\overline{\text{TEST}}$ abgefragt. Dieser Eingang dient vor allem der Zusammenarbeit mit dem numerischen Coprozessor 8087. Sein Ausgang BUSY wird mit dem $\overline{\text{TEST}}$-Eingang verbunden.

Der wohl bedeutsamste Pin ist der Eingang MN/$\overline{\text{MX}}$. Die Buchstaben stehen für Minimum und Maximum und bezeichnen die zwei grundlegenden Modi, mit denen der Prozessor 8086 betrieben werden kann. Der Eingang ist hardware-mäßig entweder mit Plus oder Masse verbunden. Wenn Masse anliegt, ist der Baustein im Maximum-Modus und den Pins 24 bis 31 ist die im Bild 2-5 nicht eingeklammerte Funktion zugewiesen. Die Funktionen, die zuvor die eingeklammerten Pins besaßen, werden nun binärkodiert an den Ausgängen S_{0-2} ausgegeben. In diesem Modus ist die CPU auf die Zusammenarbeit mit dem Buskontroller 8288 angewiesen, der die Dekodierung übernimmt.

Die Pins 30, 31 (RQ/GT) können von anderen Prozessoren benutzt werden, die in demselben System vorhanden sind. Masse bewirkt Einstellung der Prozessoraktionen und Überlassung des Busses an andere Prozessoren. Werden diese Eingänge nicht benötigt, so kann man sie wegen intern vorhandener Pull-Ups unbeschaltet lassen.

Die Ausgänge QS1, QS0 spiegeln den internen Queue-Status der Befehlsverarbeitung wieder.

High-Pegel an NMI löst einen Non Maskable Interrupt am Prozessor aus.

2.6 Der Mikroprozessor 8088

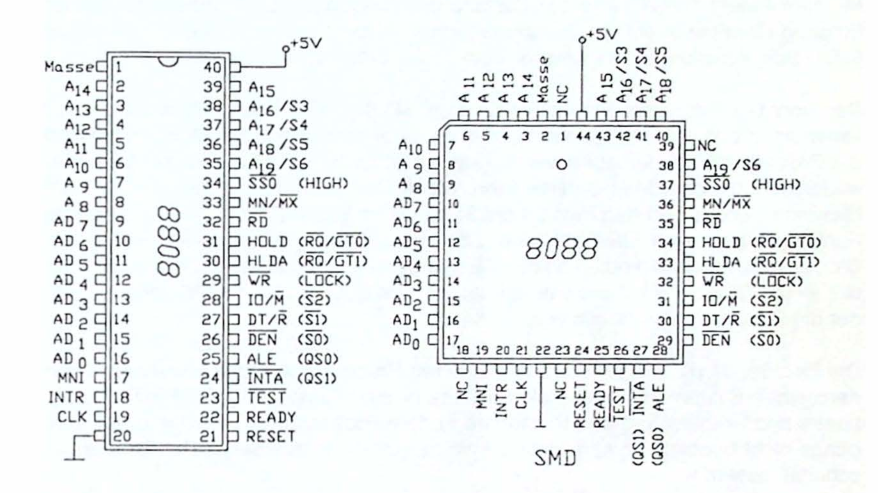


Bild 2-6. Pin-Belegung des Mikroprozessors 8088

Da der Prozessor 8088 nichts weiter als ein 8086er-Kompromiß an ein 8-Bit-Bussystem ist, unterscheidet er sich nur geringfügig von dem zuvor beschriebenen 8086. Bild 2-6 kann man entnehmen, daß der 8088 nur einen gemultiplexten 8-Bit-Adreß-/Datenbus besitzt AD_{0-7}. Somit entfällt auch die Notwendigkeit eines $\overline{BHE}$-Signals. An seiner Stelle hat der 8088 im Minimum-Modus den Ausgang $\overline{SS0}$, ein Signal, mit dessen Hilfe und in Verbindung mit den $IO/\overline{M}$- und $DT/\overline{R}$-Signal eine vollständige Dekodierung des aktuellen Busstatus möglich ist, vergleichbar den Statusleitungen S_{0-2} im Maximum-Modus. Im Maximum-Modus hat der Ausgang keine Bedeutung und liegt beständig an Plus. Man beachte, daß die in Bild 2-6 eingeklammerten Funktionen für den Maximum-Modus gelten.

Die Funktionen der restlichen Pins entnehme man der Beschreibung des Prozessors 8086.

2.7 Der Mikroprozessor 80186

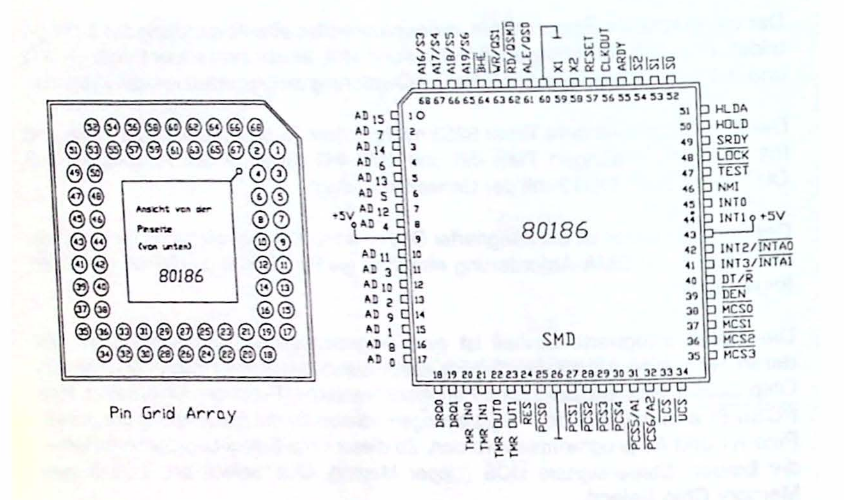


Bild 2-7. Pin-Belegung des Mikroprozessors 80186

Der 80186 ist ein 8086, in dessen Gehäuse dank fortschreitender Technik die Komponenten integriert sind, auf die ein 8086-System nicht verzichten kann. Es ist nicht übertrieben zu sagen, der 80186 stellt den größten Teil der Hauptplatine eines 8086-Systems dar. Denn in ihm sind neben dem Prozessor 15 bis 20 weitere Systembausteine der engeren Peripherie integriert. Aus dieser Integration resultiert ein doppelt so schneller Datendurchsatz. Die Software ist identisch mit dem Befehlssatz des 8086, aber wegen der integrierten Peripheriebausteine sind zehn neue Befehlstypen hinzugekommen

Die wichtigsten integrierten Bestandteile sind, wie bereits genannt, die CPU 8086, die mit den Pins AD_{0-15} und A_{16-19} den gemultiplexten Adreß-/Datenbus ausgibt.

Der CPU angegliedert ist der Buscontroller 8288, der in Verbindung mit der CPU die Signale $\overline{S}_{0\text{-}2}$, HLDA, RESET, DT/$\overline{R}$, $\overline{LOCK}$, $\overline{DEN}$, $\overline{BHE}$, $\overline{RD}$, $\overline{WR}$ und ALE ausgibt und die Signale SRDY, ARDY, $\overline{TEST}$, HOLD und $\overline{RES}$ empfängt.

Die wichtigsten Funktionen des Taktgenerators 8284 sind ebenfalls integriert. Die externen Komponenten des Oszillators werden an den Eingängen X1 und X2 angeschlossen. Die so erzeugte Frequenz liegt durch zwei geteilt mit einem Tastverhältnis von 50 Prozent am Ausgang CLKOUT an.

Der vierte Block wird durch den Interruptcontroller, eine Abwandlung des 8259, gebildet. Er besitzt fünf Eingänge INT0-4 und MNI, wovon zwei seiner Eingänge INT2 und INT3 als Ausgänge zur Interrupt-Quittierung programmiert werden können.

Der fünfte Block ist dem Timer 8253 nachgebildet. Er besitzt drei 16-Bit-Zähler und tritt mit den Eingängen TMR IN1 und TMR IN2 sowie mit den Ausgängen TMR OUT1 und TMR OUT2 mit der Umwelt in Kontakt.

Der sechste Block ist ein integrierter DMA-Kontroller mit ähnlichen Funktionen wie der 8237. Eine DMA-Anforderung wird über die Pins DRQ0 und DRQ1 am 80186 angemeldet.

Die siebte integrierte Einheit ist eine programmierbare Chip-Select-Logik, mit deren Hilfe eine Adreßdekodierung zur Auswahl bestimmter Speicher- (Memory Chip Select, Pins $\overline{MCS0\text{-}3}$) und Peripheriebausteine (Peripheral Chip Select, Pins $\overline{PCS0\text{-}6}$) erfolgt. Zwei der $\overline{PCS}$-Leitungen können für die Speicherung der Adreß-Pins A1 und A2 programmiert werden. Zu dieser Chip-Select-Logik gehören ferner die beiden Steuersignale $\overline{UCS}$ (Upper Memory Chip Select) und $\overline{LCS}$ (Lower Memory Chip Select).

2.8 Der Mikroprozessor 80188

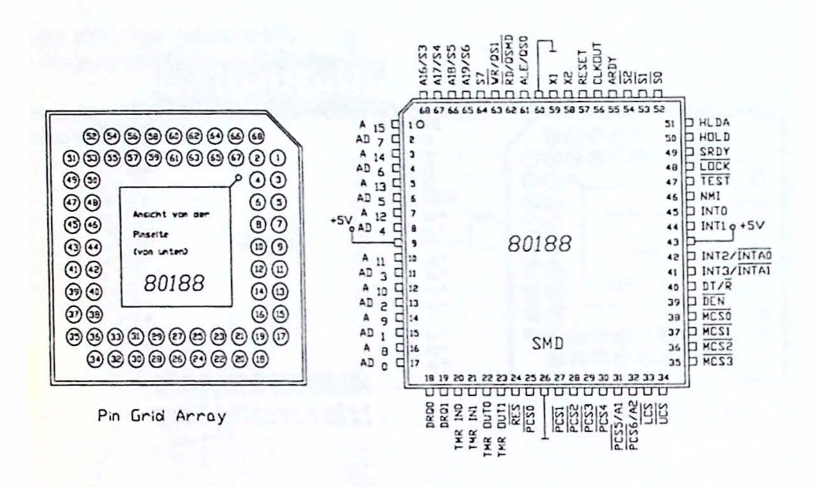


Bild 2-8. Pin-Belegung des Mikroprozessors 80188

Der 80188 ist die 8-Bit-Datenbus-Variante des 80186. Wie der Prozessor 8088 sich vom 8086 unterscheidet, so unterscheidet sich der 80188 vom 80186. Er verfügt über einen gemultiplexten 8-Bit-Adreß-/Datenbus, der über die Pins AD_{0-7} ausgegeben wird. Ansonsten sind die integrierten Bestandteile und Funktionen dieselben, wie sie der 80186 aufweist.

2.9 Der Mikroprozessor 80286

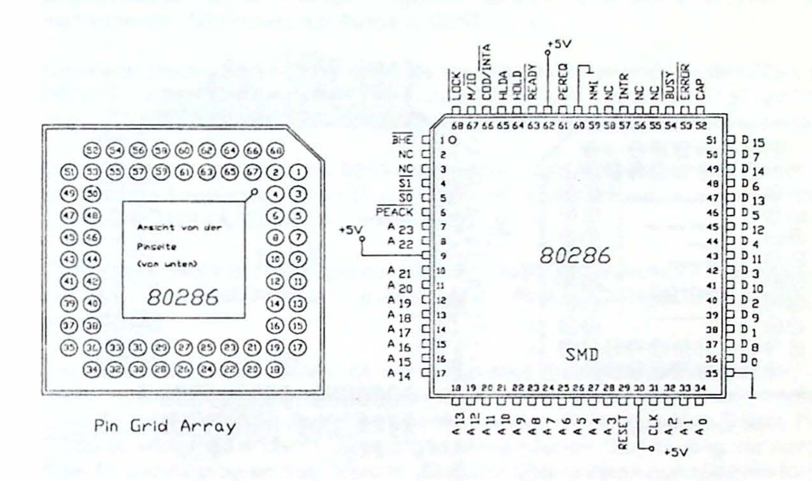


Bild 2-9. Pin-Belegung des Mikroprozessors 80286

Der 80286 ist ein sehr leistungsfähiger 16-Bit-Mikroprozessor, dessen wohl bekannteste Einsatzmöglichkeit die als Zentralprozessor in einem AT-Computer ist. Seine Leistungsfähigkeit kann bis zu sechsmal höher sein als die des 8086. Ein Blick auf seine Pin-Belegung (Bild 2-9) läßt erkennen, daß er über keinen gemultiplexten Adreß-/Datenbus verfügt. Er stellt 24 Adreßleitungen zur Verfügung und kann somit 16 MByte adressieren. Mit Hilfe des integrierten Memory Managements ist sogar die Verwaltung von 1 GByte ($=2^{30}$ Byte) Adreßbereich möglich. Zur Steuerung des Adreßflusses besitzt der 80286 die Signale $\overline{BHE}$ und M/$\overline{IO}$. Der I/O-Bereich umfaßt 64 K zu 8 Bit oder 32 K zu 16 Bit.

Die Daten fließen über die Datenpins D_{0-15}, die gleichzeitig den Datenbus bilden. Über ihn kann der 80286 einzelne Bytes oder Wörter handhaben. Zur Differenzierung von Bytes und Wörter werden die Signale A_0 und $\overline{BHE}$ verwendet.

Zur Steuerung von Transceiver und zur Erzeugung von Schreib-/Lesesignalen ist der Buscontroller 82288 notwendig. An ihn werden die Signale $\overline{S0}$, $\overline{S1}$ und M/$\overline{IO}$ geführt, die er dekodiert und die Systemverwaltungssignale erzeugt. Insgesamt verfügt der 80286 über acht Status- und Kontrollsignale.

Es existieren zwei Modi, mit deren Hilfe der 80286 zum 8086 abwärts kompatibel ist:

- der 8086-Real-Adreßmodus,
- der geschützte virtuelle Adreßmodus.

Im ersten Modus unterstützt der 80286 nur denselben Adreßbereich wie der 8086, nämlich 1 MByte.

2.10 Der Mikroprozessor 80386

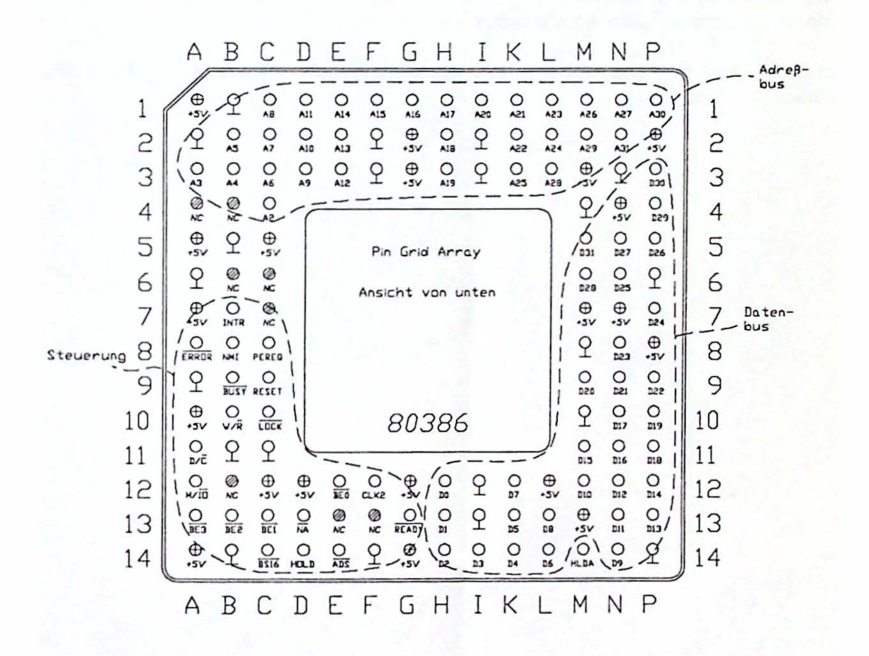


Bild 2-10 a. Pin-Belegung des Mikroprozessors 80386

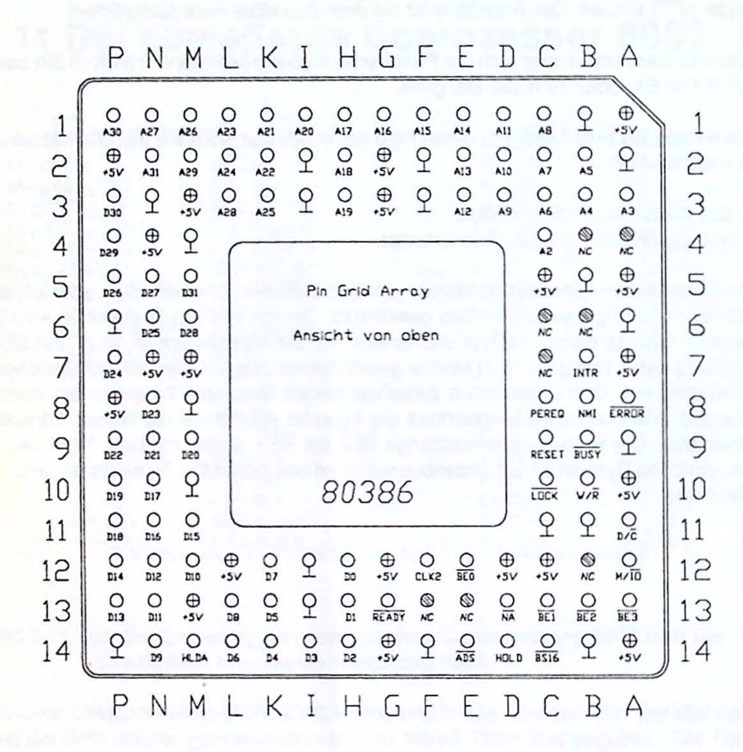


Bild 2-10 b. Pin-Belegung des Mikroprozessors 80386

Der 80386 besteht aus einer CPU, einer Speicherverwaltungseinheit und einem Bus-Interface. Er ist ein 32-Bit-Mikroprozessor, der aus Kompatibilitätsgründen Datentransfers von acht, 16 und 32 Bitbreiten unterstützt. Die Wörter werden an zwei aufeinanderfolgenden Bytes im Speicher abgelegt, wobei dem LSB die nied-

rigste Adresse zukommt. Doppelwörter werden mit vier Bytes in entsprechender Weise abgelegt. Die Speichergröße beträgt 4 GigaByte physikalisch und 64 TeraByte (2^{46}) virtuell. Der Adreßbus ist mit dem Datenbus nicht gemultiplext.

Der I/O-Bereich, in dem sich die Peripheriebausteine befinden, ist 64 K (8 Bit) oder 32 K (16 Bit) oder 16 K (32 Bit) groß.

Es existieren zwei Modi, mit deren Hilfe der 80386 zur Software des 8086 abwärts kompatibel ist:

- der 8086-Real-Adreßmodus,
- der geschützte virtuelle Adreßmodus.

Im Real-Modus kann der 80386 wie ein sehr schneller 8086 arbeiten, aber mit einer Erweiterung von 32 Bit, falls gewünscht. Der zur Verfügung stehende Adreßbereich beträgt dann 1 MByte und es sind nur die Adreßleitungen A_{2-19} und BE0 bis BE3 aktiv. Der geschützt Modus gewährleistet Zugang zum offenen Speicher-Management. Der Unterschied zwischen beiden Modi liegt hauptsächlich darin, wie die Speicherverwaltungseinheit die logische Adresse in die lineare Adresse übersetzt. Die Byte-Enable-Ausgänge $\overline{BE0}$ bis $\overline{BE3}$ zeigen mit Low-Pegel direkt an, welches Byte im 32-Bit-Datenbus während des laufenden Transfers angesprochen ist.

2.11 Der numerische Coprozessor 8087

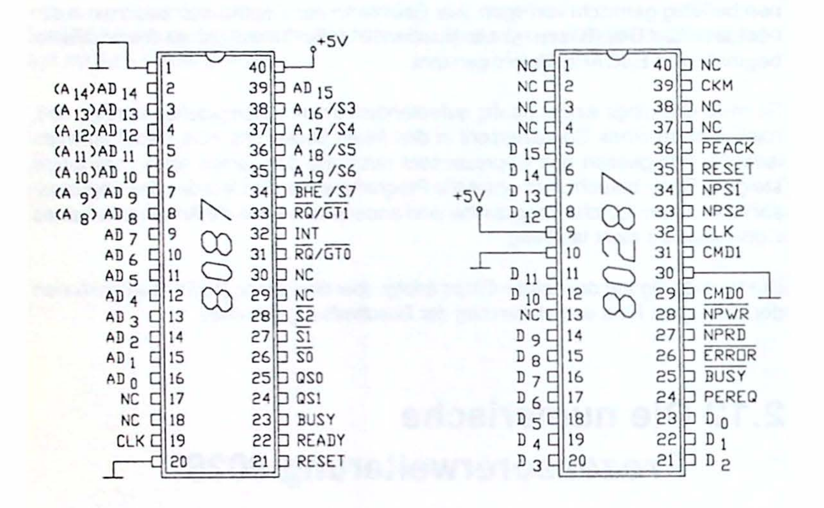


Bild 2-11. Die Pin-Belegung des numerischen Coprozessors 8087 und der numerischen Prozessorerweiterung 80287

Als Coprozessor für einen 8086, 8088, 80186 und 80188 wird der 8087 parallel neben der CPU auf den gemultiplexten lokalen Adreß-/Datenbus gegeben. Die Daten-/Adreßleitungen der CPU und des Coprozessors sind direkt miteinander verbunden. Daher vermittelt das Bild der Pin-Belegung (Bild 2-11) einen bekannten Eindruck (siehe Bild 2-5, Bild 2-6). Die gemultiplexten Busleitungen werden durch die Pins AD_{0-15} dargestellt. Nur in Zusammenarbeit mit den Prozessoren 8086 und 80186 müssen beide Adreß-Bytes zwischengespeichert werden; in Zusammenarbeit mit den Prozessoren 8088 und 80188 ist eine Speicherung nur des Low-Bytes vonnöten, da das High-Byte der Adresse während des gesamten Speicher- oder I/O-Zugriffs beständig an den Ausgängen A_{8-15} anliegt. Der Coprozessor bestimmt unmittelbar nach dem Reset durch Überwachung der $\overline{BHE}$/S7-Leitung, ob die CPU ein 8086/186 oder 8088/188 ist. Während der Operation bewirkt der ständige Kontakt mit der CPU über die Leitungen QS0, QS1 die nötige Synchronisation.

Für den Programmierer erscheinen CPU und Coprozessor als eine Einheit. CPU und Coprozessor lesen die eingehenden Befehle immer gemeinsam und erkennen am Operationskode, ob es ein Befehl für die CPU oder den Coprozessor ist. Vereinfacht ausgedrückt sind CPU-Befehle NOPs (No Operation) für den Coprozessor und Coprozessorbefehle NOPs für die CPU, d.h. die beiden Befehlsarten können beliebig gemischt vorliegen. Alle Befehle für den Coprozessor besitzten in den höchsten fünf Bits (Bits_{11-15}) das Muster 11011. Ein Befehl, der mit diesem Muster beginnt, wird ESCAPE-Befehl genannt.

Es muß allerdings einem häufig auftretendem Irrtum widersprochen werden, daß, nach Einbau eines Coprozessors in den freien Sockel des PCs, dieser automatisch die Fähigkeiten des Coprozessors nutzt und das Tempo eines Programms steigert. Dazu braucht man spezielle Programme, die die Opcodes des Coprozessors benutzen. Solche Programme sind andererseits ohne die Anwesenheit eines Coprozessors nicht lauffähig.

Die Verständigung der beiden Chips erfolgt über die Leitung BUSY. Die Funktionen der restlichen Pins entnehme man der Beschreibung des 8086.

2.12 Die numerische Prozessorerweiterung 80287

Im üblichen Sprachgebrauch wird auch dieser Baustein als mathematischer Coprozessor bezeichnet. Beim Blick auf seine Pin-Belegung (Bild 2-11) fällt schon auf, daß er viele Pins ohne Funktion (NC, Not Connected) besitzt. Beim zweiten Blick vermißt man die Adreßleitungen, lediglich der 16-Bit breite Datenbus und diverse Steuerleitungen sind vorhanden.

Der 80287 ist für eine Zusammenarbeit mit dem 8086 konzipiert, der über einen getrennten Adreß- und Datenbus verfügt. Anders als der 8087 wird der 80287 von der CPU wie ein unmündiges Kind behandelt. Die CPU dekodiert für ihn die Befehle, selektiert den 80287 über die Eingänge $\overline{\text{NPS1}}$ und NPS2 (Numeric Prozessor Selects) und veranlaßt ihn ähnlich wie ein Peripheriebaustein über die Eingänge $\overline{\text{NPRD}}$ (Numeric Prozessor Read) und $\overline{\text{NPWR}}$ (Numeric Prozessor Write), die für ihn geltenden Befehle und Daten vom Bus zu lesen bzw. zu schreiben.

Der Befehlssatz ist mit dem des 8087 identisch.

Bei der Verwendung des 80287 in einem 80286-System sind die beiden Datenleitungen miteinander verbunden. Ferner sind die folgenden gleichnamigen Pins der CPU und des Coprozessors miteinander verbunden: $\overline{\text{ERROR}}$, $\overline{\text{BUSY}}$, $\overline{\text{PEACK}}$, PEREQ. Über den Buscontroller 82288 werden die Eingänge $\overline{\text{NPDR}}$ und $\overline{\text{NPWR}}$ angesteuert.

Der 80287 kann auch als eine numerische Prozessorerweiterung in Zusammenarbeit mit dem 80386 dienen.

2.12 Die numerische Prozessorerweiterung 8087

Kapitel 3

3. Bausteine der engeren Prozessorperipherie

Die Bausteine, die in diesem Kapitel vorgestellt werden, erfüllen entweder Aufgaben, die der allgemeinen Busverwaltung dienen wie die Chip-Select-Dekoder oder die Busverwalter, oder sind für die Funktion der Prozessoren unabdingbar (Taktgeneratoren oder Interruptcontroller). Letzere sind für eine Zusammenarbeit für ganz bestimmte Prozessoren optimiert, können aber mit kleinen Abwandlungen in der Schaltung oder kleinen Einschränkungen mit allen in Kapitel 2 vorgestellten Prozessoren zusammenarbeiten.

3.1 Die programmierbaren Chip-Select-Dekoder 82C138/139, 82C338/339

3.1.1 Beschreibung

Bei diesen Bausteinen handelt es sich um maskenprogrammierbare Chip-Auswahlbausteine in CMOS-Technologie, die von der CPU für die Selektierung von Speicher bzw. Peripherie-Chips verwendet werden können. Sie gleichen in ihrer Funktion den Standardbausteinen 74HC138/74HC139, sind ihnen jedoch in so mancher Hinsicht überlegen. Verglichen mit den Bausteinen der 74HCxx-Reihe erfolgt eine Geschwindigkeitssteigerung um den Faktor drei bis vier. Alle Adreßeingänge können mit einem ALE-Signal zwischengespeichert werden, so daß die

Bausteine sowohl in einem gemultiplexten Adreß-/Datenbus als auch in einem nicht gemultiplexten Adreßbus (mit ALE an Plus) betrieben werden können. Durch ihre Programmierbarkeit können sie in fast jedes Prozessorensystem eingegliedert werden.

Die an den Eingängen A bis C, LH, LB und Gx (sofern vorhanden) anliegende Adresse wird mit der negativen Flanke am ALE-Eingang intern aufbewahrt. Mit Masse am SEL-Eingang ist der Chip selektiert, d.h. der der Adresse entsprechende Yx-Ausgang liegt an Masse, die restlichen an High-Pegel. Mit SEL an Plus führen alle Yx-Ausgänge High-Pegel, sie sind nicht im Tri-State. Um bei Adreßwechsel eine Fehlinformation an den Ausgängen zu vermeiden, verbindet man häufig den SEL- mit dem ALE-Eingang.

Der Ausgang MATCH ist mit der Ausnahme des 82C338 bei allen Bausteinen vorhanden. Der Zustand der Gx-Eingänge in Verbindung mit dem programmierten Wert bestimmt den logischen Pegel des MATCH-Ausgangs. Sind die Gx-Eingänge in Übereinstimmung mit dem programmierten Feld, geht der MATCH-Ausgang unabhängig vom Zustand des SEL-Eingangs an Masse (aktiver Zustand), gleichzeitig wird der den A- bis C-Eingängen entsprechende Yx-Ausgang freigegeben. Sind die Gx-Eingänge von dem programmierten Muster verschieden, bleiben MATCH- und Yx-Ausgänge an High-Pegel. Der MATCH-Ausgang dient hauptsächlich für Busverwaltungszwecke oder der Kontrolle über Wait-States.

Die Gx-Eingänge sind auf den high-aktiven bzw. low-aktiven Zustand programmierbar. Die High- und Low-Bank-Auswahleingänge (HB, LB) in den 82Cxx9 Bausteinen sind in gleicher Weise programmierbar. Das bewirkt einen optimalen Einsatz in 8-Bit- oder 16-Bit-orientierten Systemen. Durch die Fuse-Link-Technik ist in diesen Bausteinen die Information permanent und sicher über den gesamten Temperaturbereich eingebrannt.

Ist beispielsweise im Baustein 82C138 für die G5- bis G1-Eingänge das Bitmuster 1011 0 programmiert und werden in einem 64-K-System die Adreßleitungen nach folgendem Schema mit den Eingängen verbunden,

Adreßleitungen:	A15	A14	A13	A12	A11	A10	A9
Eingänge (82C138):	G5	G4	G3	G2	G1	B	A

dann sind die Yx-Ausgänge nach folgender Tabelle freigegeben:

Ausgang	MSB	Adreßbereich im 64-K-Block
Y0	1011 0000 und 1011 0001	B000 - B1FF
Y1	1011 0010 und 1011 0011	B200 - B3FF
Y2	1011 0100 und 1011 0101	B400 - B5FF
Y3	1011 0110 und 1011 0111	B600 - B7FF

Tabelle 3-1. Fensteröffnungen im 64-K-Block mit dem 83C138 und einer Beispielsprogrammierung

Somit stehen vier Fenster in einem 64-K-Block mit einer jeweiligen Größe von einem halben KByte an den Adressen B000 bis B7FF zur Verfügung.

3.1.2 Die Bausteine 82C138 und 82C139

In typischen Anwendungen können der 82C138 und 82C139 zwei bis drei Bausteine der 74HC-Serie ersetzen. Sie sind geeignet für 8- bzw. 16-Bit-Prozessoranwendungen als "Bootstrap"-PROM-Dekoder oder anderen Speicher- und I/O-Dekoderanwendungen, wo vier Bausteine oder weniger in einem Adreßbereich selektiert werden müssen. Der 82C138 besitzt intern einen 2-zu-4-Dekoder mit vier Ausgängen, der 82C139 lediglich einen 1-zu-2-Dekoder. Durch die Bank-Select-Eingänge LB und HB ist letzterer auf zwei Bankausgänge umschaltbar. Die LB- bzw. LH-Eingänge werden einem Exklusiv-Oder-Gatter zugeführt, dessen zweiter Eingang auf einen logischen Pegel programmierbar ist (Bild 1-3). Ist Low-Pegel programmiert, bewirkt High-Pegel am LB-Pin ebenfalls High-Pegel am Ausgang und die Ausgänge Y0 oder Y1 können vom 1-zu-2-Dekoder ausgewählt werden. Wenn High-Pegel programmiert wurde, ist der Eingang low-aktiv. In gleicher Weise verhält sich der High-Bankeingang (HB).

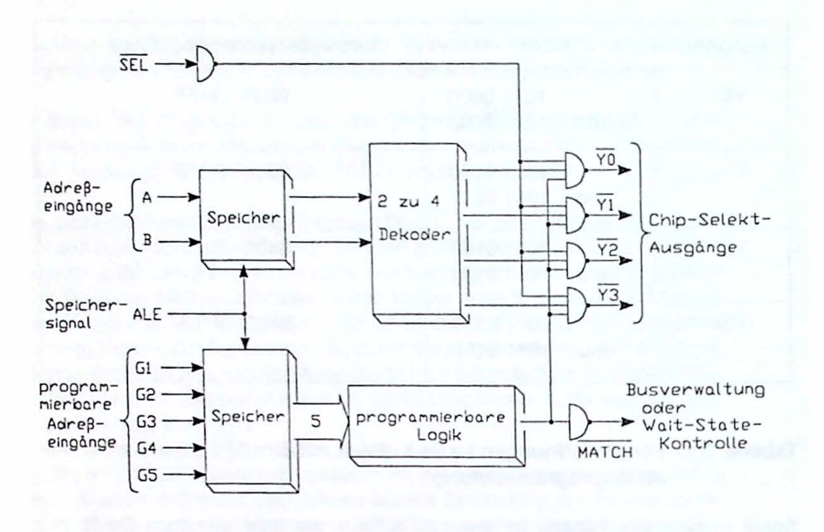


Bild 3-1. Blockschaltbild des programmierbaren Dekoders 82C138

Eingänge				Ausgänge				
A	B	SEL	Gx	Y0	Y1	Y2	Y3	MATCH
0	0	0	W	0	1	1	1	0
0	1	0	W	1	0	1	1	0
1	0	0	W	1	1	0	1	0
1	1	0	W	1	1	1	0	0
X	X	1	W	1	1	1	1	0
X	X	X	F	1	1	1	1	1

0 : Low-Pegel
1 : High-Pegel
X : Low- oder High-Pegel
W: Kode in Übereinstimmung mit Programmierung
F : Kode nicht in Übereinstimmung mit Programmierung

Tabelle 3-2. Wahrheitstafel des programmierbaren Dekoders 82C138

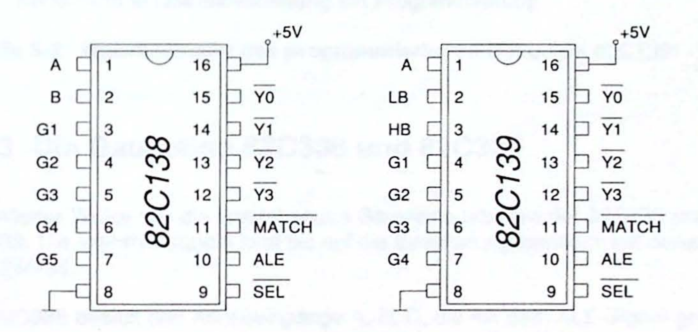


Bild 3-2. Pin-Belegung der programmierbaren Dekoder 82C138 und 82C139

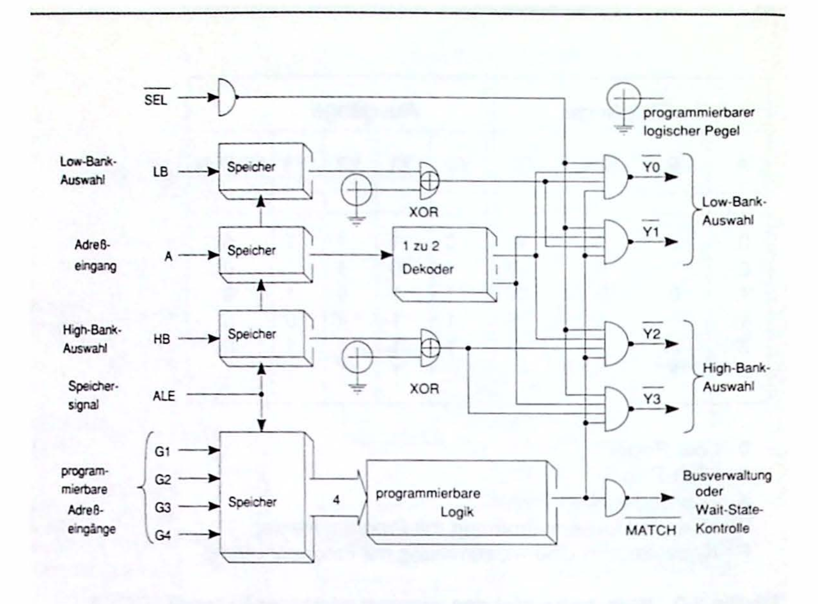


Bild 3-3. Blockschaltbild des programmierbaren Dekoders 82C139

Eingänge					Ausgänge				
A	LB	HB	SEL	Gx	Y0	Y1	Y2	Y3	MATCH
0	W	F	0	W	0	1	1	1	0
1	W	F	0	W	1	0	1	1	0
0	F	W	0	W	1	1	0	1	0
1	F	W	0	W	1	1	1	0	0
0	F	F	0	W	1	1	1	1	0
1	F	F	0	W	1	1	1	1	0
0	W	W	0	W	0	1	0	1	0
1	W	W	0	W	1	0	1	0	0
X	X	X	1	W	1	1	1	1	0
X	X	X	X	F	1	1	1	1	1

0 : Low-Pegel
1 : High-Pegel
X : Low- oder High-Pegel
W: Kode in Übereinstimmung mit Programmierung
F : Kode nicht in Übereinstimmung mit Programmierung

Tabelle 3-3. Wahrheitstafel des programmierbaren Dekoders 82C139

3.1.3 Die Bausteine 82C338 und 82C339

In ähnlicher Weise wie die vorstehenden Bausteine arbeiten der 82C338 und der 82C339. Die Wahrheitstafeln sind bis auf die Erweiterung identisch mit denen der 82C138/139.

Der 82C338 besitzt drei Adreßeingänge A, B, C, die mit dem ALE-Signal gespeichert werden können und die anschließend einem 3-zu-8-Dekoder zugeführt werden. Die programmierbaren Adreßeingänge G1 bis G5 werden ebenfalls mit dem ALE-Signal zwischengespeichert und ihre Pegel mit den programmierten Werten verglichen. Daraufhin werden die dekodierten Ausgänge Y0 bis Y7 freigegeben oder gesperrt. Einen MATCH-Ausgang gibt es in diesem Baustein nicht. Mit High-Pegel am SEL-Eingang können ebenfalls die Yx-Ausgänge gesperrt werden.

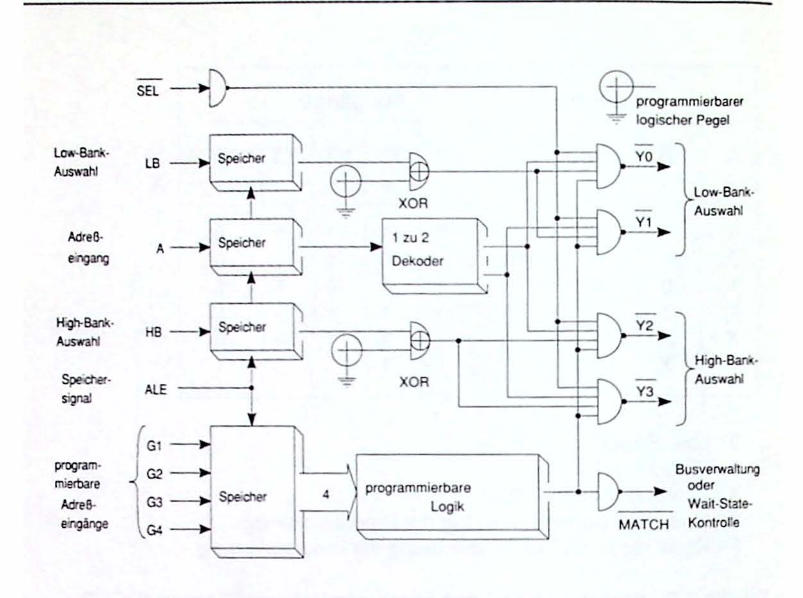


Bild 3-3. Blockschaltbild des programmierbaren Dekoders 82C139

Eingänge					Ausgänge				
A	LB	HB	SEL	Gx	Y0	Y1	Y2	Y3	MATCH
0	W	F	0	W	0	1	1	1	0
1	W	F	0	W	1	0	1	1	0
0	F	W	0	W	1	1	0	1	0
1	F	W	0	W	1	1	1	0	0
0	F	F	0	W	1	1	1	1	0
1	F	F	0	W	1	1	1	1	0
0	W	W	0	W	0	1	0	1	0
1	W	W	0	W	1	0	1	0	0
X	X	X	1	W	1	1	1	1	0
X	X	X	X	F	1	1	1	1	1

0 : Low-Pegel
1 : High-Pegel
X : Low- oder High-Pegel
W: Kode in Übereinstimmung mit Programmierung
F : Kode nicht in Übereinstimmung mit Programmierung

Tabelle 3-3. Wahrheitstafel des programmierbaren Dekoders 82C139

3.1.3 Die Bausteine 82C338 und 82C339

In ähnlicher Weise wie die vorstehenden Bausteine arbeiten der 82C338 und der 82C339. Die Wahrheitstafeln sind bis auf die Erweiterung identisch mit denen der 82C138/139.

Der 82C338 besitzt drei Adreßeingänge A, B, C, die mit dem ALE-Signal gespeichert werden können und die anschließend einem 3-zu-8-Dekoder zugeführt werden. Die programmierbaren Adreßeingänge G1 bis G5 werden ebenfalls mit dem ALE-Signal zwischengespeichert und ihre Pegel mit den programmierten Werten verglichen. Daraufhin werden die dekodierten Ausgänge Y0 bis Y7 freigegeben oder gesperrt. Einen MATCH-Ausgang gibt es in diesem Baustein nicht. Mit High-Pegel am SEL-Eingang können ebenfalls die Yx-Ausgänge gesperrt werden.

Der 82C339 besitzt zwei Adreßeingänge A und B, zwei Eingänge, die Low- bzw. High-Bank auswählen sowie sieben programmierbare Adreßeingänge G1 bis G7. Alle Eingänge bis auf den SEL-Eingang können mit dem ALE-Signal intern zwischengespeichert werden. In seiner Funktion ist er vergleichbar dem 82C139, unterscheidet sich aber von ihm durch eine größere Zahl von Ein- bzw. Ausgängen.

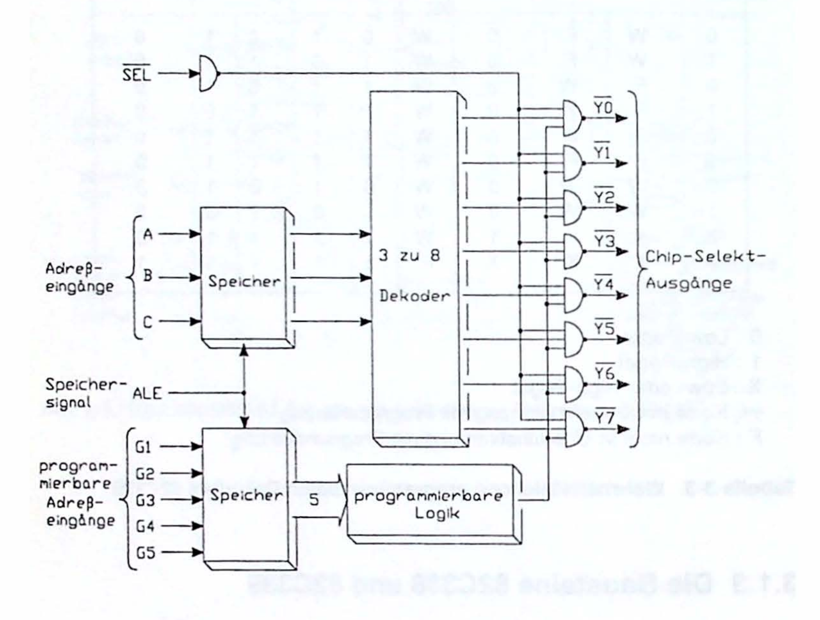


Bild 3-4. Blockschaltbild des programmierbaren Dekoders 82C338

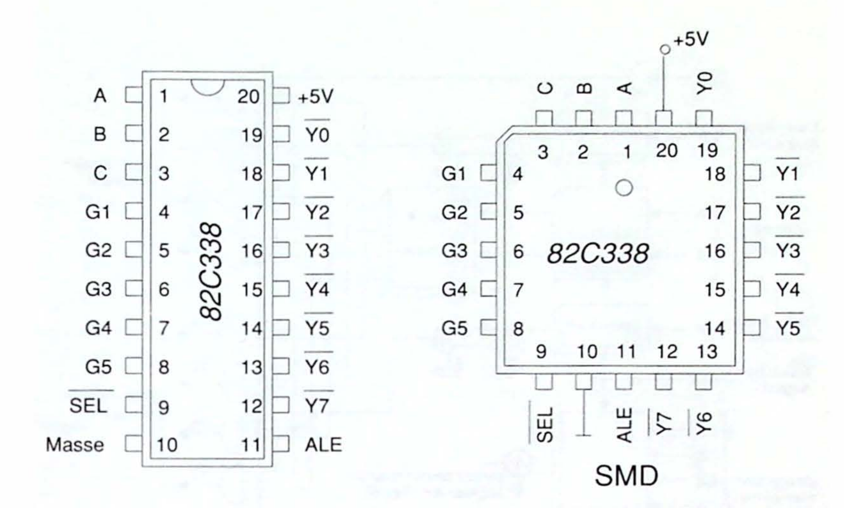


Bild 3-5. Pin-Belegung des programmierbaren Dekoders 82C338

Der 82C339 befindet sich in einem 24poligem Gehäuse, da er zwei Bank-Select-Eingänge (LB, HB), zwei Adreßeingänge (A, B), sieben programmierbare Adreßeingänge (G1 - G7), einen ALE- und einen Select-Eingang, sowie neun Ausgänge besitzt. Er stellt im Prinzip einen erweiterten 82C139 mit den gleichen Eigenschaften dar.

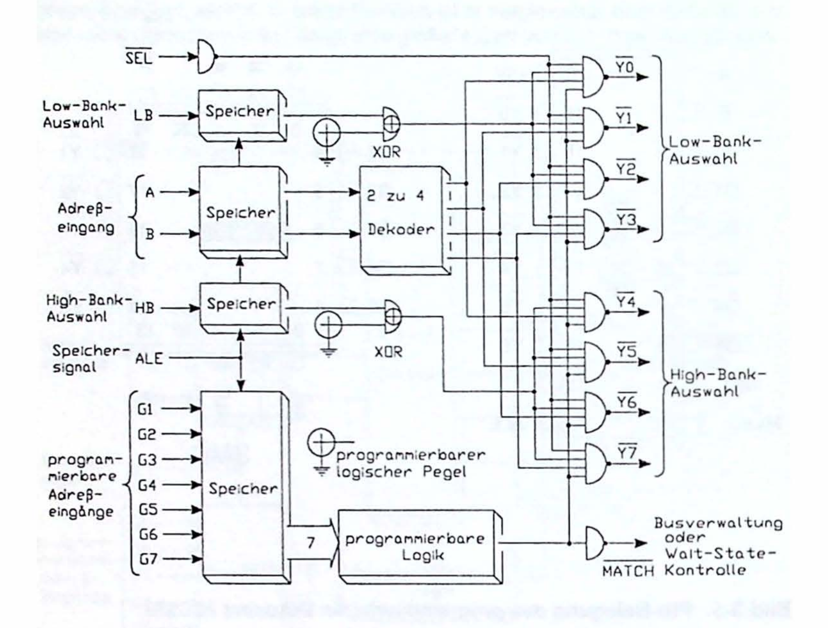


Bild 3-6. Blockschaltbild des programmierbaren Dekoders 82C339

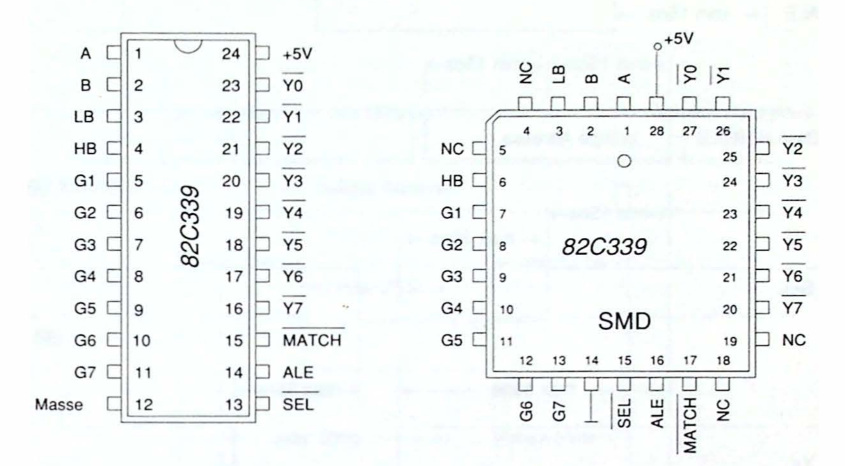


Bild 3-7. Pin-Belegung des programmierbaren Dekoders 82C339

Für eine funktionierende Zusammenarbeit mit dem Bussystem ist die Kenntnis über das zeitliche Verhalten der Bausteine wichtig. Dieses Timing ist für alle vier Chips das gleiche. In den Bildern 3-8 und 3-9 sind die minimalen und maximalen Zeiten genannt. Diese Zeiten sollten nicht unter- bzw. überschritten werden, um eine Überschneidung der Yx-Ausgänge zu vermeiden. Der SEL-Eingang kontrolliert lediglich die Yx-Ausgänge, nicht aber den MATCH-Ausgang.

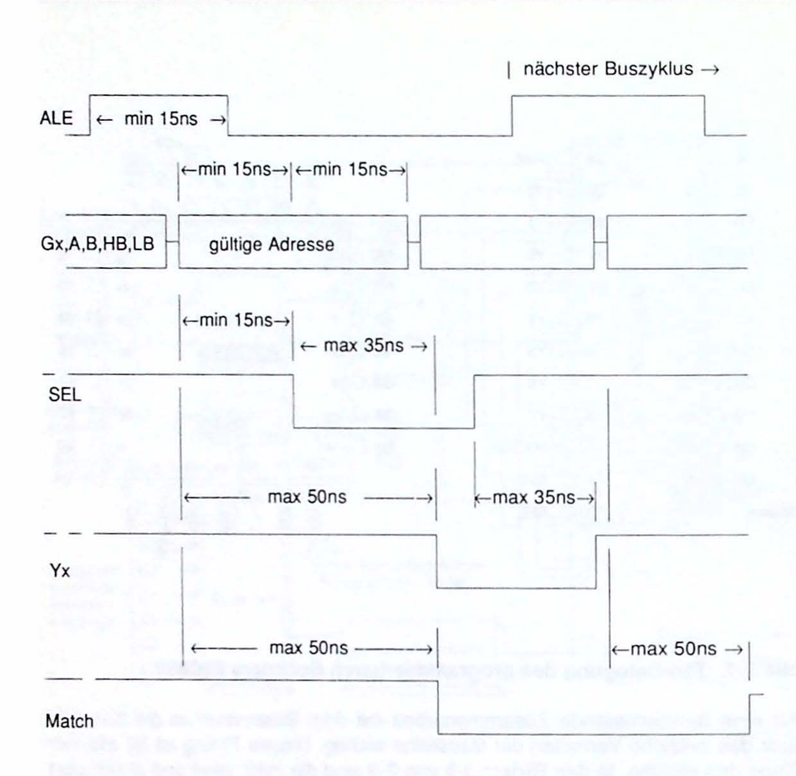
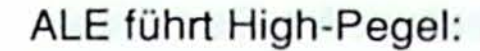


Bild 3-8. Taktsignale der gemultiplexten Busoperation



ALE führt High-Pegel:

|←min 15ns→|←min 15ns→| |→ nächster Buszyklus

Gx,A,B,HB,LB gültige Adresse

|←min 15ns→| |← min 15 ns →|

← max 35ns →

SEL

←—— max 50ns ——→ |←max 35ns→|

Yx

←—— max 50ns ——→ |←max 50ns—→|

Match

Bild 3-9. Taktsignale der nicht gemultiplexten Busoperation

3.1.4 Die Programmierung

Zum Programmieren der Bausteine müssen interne Sicherungen durchgebrannt werden. Beim Kauf sind alle Sicherungen intakt. Man kann sich ein Programmiergerät bauen, das den nachstehenden Forderungen gerecht wird, oder auf kommerziell erhältliche Geräte zurückgreifen. Bezugsquellennachweise sind über Firma Matra-Harris zu erhalten. (Die Adresse findet sich in 4.6.8.)

Die Programmierung erfolgt bei allen Bausteinen nach derselben Art. Es müssen folgende Punkte beachtet werden:

3.1.4.1 Schutzmaßnahmen

- Bei der Programmierung oder bei Betrieb darf kein Eingangs-Pin in der Luft hängen.
- Die Eingangsspannung darf an keinem Pin die positive Betriebsspannung überschreiten.
- Jeder Baustein sollte mit einem Kondensator der Kapazität 0,1 µF entkoppelt werden, der in der Nähe des Chips und zwischen Plus und Minus angebracht sein sollte.

3.1.4.2 Anlegen der Spannung

- Zu Beginn sollten alle Pins einschließlich der Stromversorgungseingänge an Masse liegen.
- In der Regel werden die Eingänge (außer SEL) durch einen Treiber mit offenem Kollektor angesteuert. Sie müssen daher mit Pull-Up-Widerständen nach Plus versehen werden, so daß sie automatisch den Spannungspegel annehmen, der als High-Pegel definiert ist. Das verhindert ein Überschreiten der positiven Betriebsspannung an den Eingangs-Pins und somit eine Zerstörung des Bausteins.
- Nun kann die Spannung auf +5 V ansteigen. Die Eingänge liegen über die Widerstände an High-Pegel.

3.1.4.3 Der Brennvorgang

- Nach einer Mindestzeit von 1 µs wird der Programmiermodus durch Anlegen einer Spannung von -6 V an Pin 1, dem Programmierungsfreigabe-Pin, eingeleitet. Pin 1 muß während des gesamten Programmiervorgangs an dieser Spannung verbleiben.

- Nach einer weiteren Zeit von mindestens 1 µs wird die positive Betriebsspannung und damit der Pegel der Eingänge auf den Wert von +12 V angehoben. Gleichzeitig wird der SEL-Pin je nach zu programmierendem Wert an +12 V oder an Masse gelegt. Jeder programmierbare Eingang muß einzeln mit dem gewünschten Pegel gebrannt werden. Ob eine Eins oder eine Null gesetzt wird, entscheidet der logische Pegel am SEL-Eingang. Liegt der SEL-Eingang an +12 V, wird eine Eins gebrannt, mit SEL an Masse eine Null.

- Nach mindestens 1 µs muß der zu programmierende Eingang, gleich welcher Pegel gebrannt werden soll, für die Zeit von 5 ms an Masse gezogen werden. Man rufe sich in Erinnerung, daß immer nur höchstens ein Eingang programmiert werden kann.

- Nach einer Wartezeit von mindestens 1 µs wird die positive Betriebsspannung wieder auf +5 V reduziert.

- Für die restlichen Eingänge werden die vorstehenden Schritte des Brennvorgangs mit der gewünschten Polarität am SEL-Eingang wiederholt.

- Wenn alle diese Schritte ausgeführt wurden, gebe man die negative Spannung an Pin 1 an Masse.

3.1.4.4 Die Verifizierung

- Durch Anlegen einer Testadresse kann die richtige Programmierung und das sichere Arbeiten des Chips überprüft werden.
- Das sichere Durchbrennen der internen Sicherungen kann dadurch getestet werden, daß bei der üblichen Betriebsspannung (+5 V) die Leistungsaufnahme des Chips gemessen wird. Zu diesem Zweck lege man die Eingänge an Low- oder High-Pegel und lasse die Ausgänge unbelastet, also frei. Fließt nun ein Strom von weniger als 50 µA in den Baustein, sind alle Sicherungen zuverlässig durchgebrannt. (Ein unprogrammierter Chip sollte nicht unter Spannung gesetzt werden, da die nicht durchgebrannten Sicherungen intern einen Kurzschluß erzeugen.) Das garantiert einwandfreie logische Pegel und hohe Zuverlässigkeit für die Lebenszeit des Chips. Die Bausteine, die diesen Test nicht bestehen, müssen nachgebrannt werden, um interne leistungszehrende Ströme zu vermeiden. Daraus ergibt sich, daß alle Eingänge programmiert werden müssen, unabhängig von der gewünschten High- oder Low-Polarität der Eingänge.

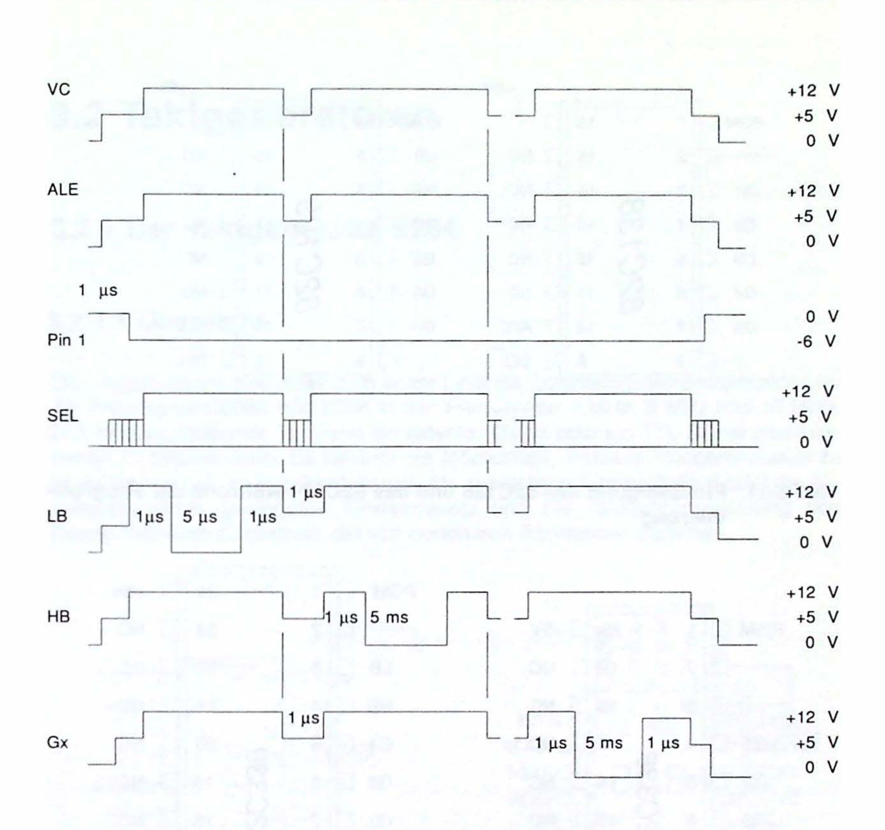


Bild 3-10. Programmierungssignale

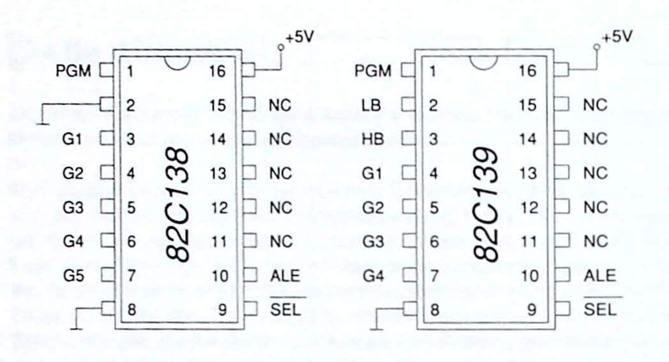


Bild 3-11. Pin-Belegung des 82C138 und des 82C139 während der Programmierung

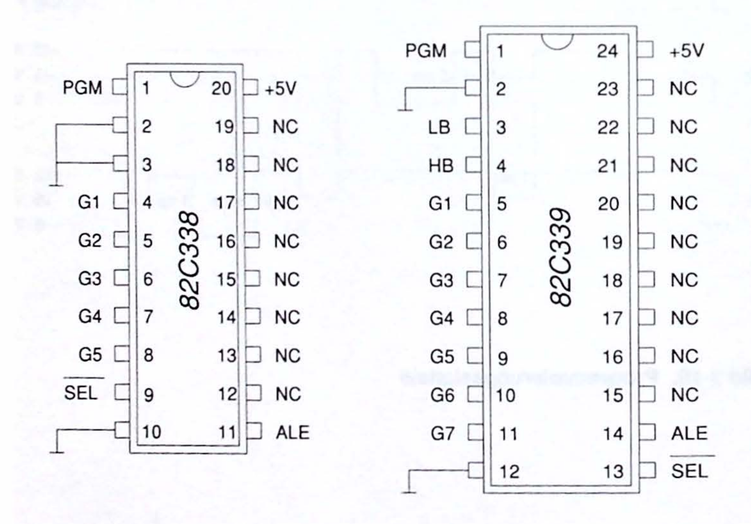


Bild 3-12. Pin-Belegung der 82C338 und 82C339 während der Programmierung

3.2 Taktgeneratoren

3.2.1 Der Taktgenerator 8284

3.2.1.1 Übersicht

Die Hauptaufgabe des 8284 ist in erster Linie die Erzeugung des Systemtaktes für die Prozessoren 8086 und 8088 in den Frequenzen 5 MHz, 8 MHz und 10 MHz. Das frequenzgebende Teil kann ein externer Quarz oder ein TTL-Signal eines externen Oszillators sein. Es besteht die Möglichkeit, mehrere Taktgeneratoren zu kaskadieren und zu synchronisieren. Als zusätzliche Eigenschaften sind die Erzeugung eines geeigneten Systemresets und die Taktsynchronisierung von Ready-Signalen zu nennen, die von peripheren Bausteinen stammen.

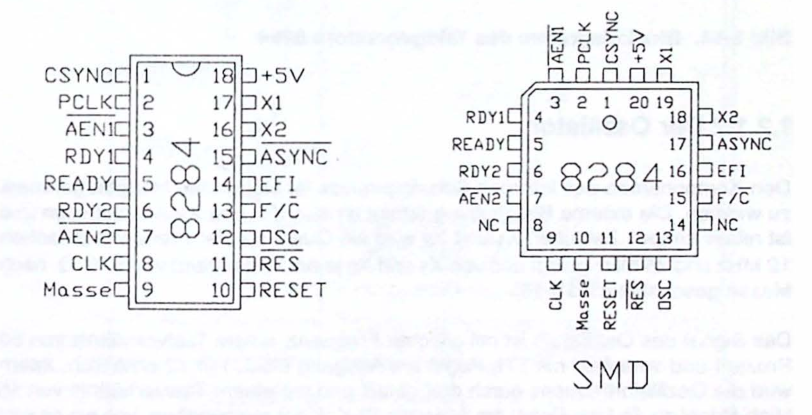


Bild 3-13. Pin-Konfiguration des Taktgenerators 8284

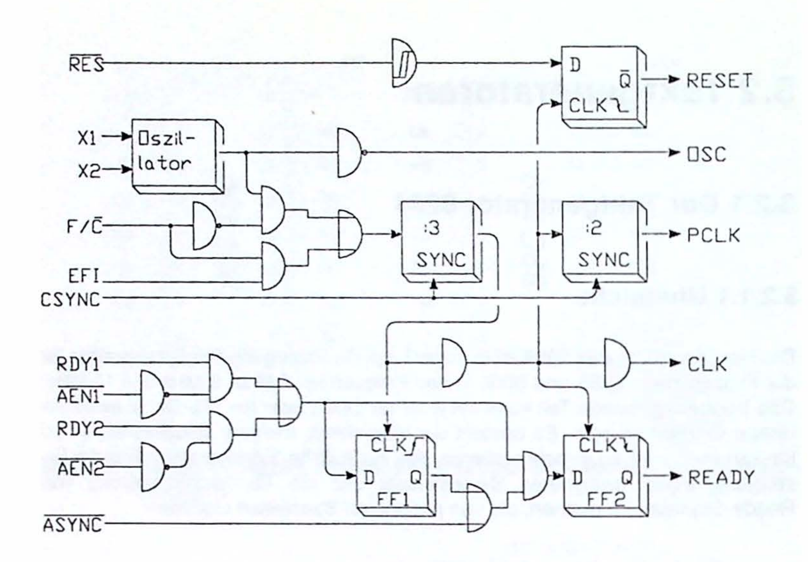


Bild 3-14. Blockdiagramm des Taktgenerators 8284

3.2.1.2 Der Oszillator

Den Komponenten des internen Schwingkreises ist sicher das Hauptaugenmerk zu widmen. Die externe Beschaltung erfolgt an den X_1- und X_2-Anschlüssen und ist relativ simpel. Zwischen X_1 und X_2 wird ein Quarz mit der Frequenz zwischen 12 MHz und 25 MHz gelegt und von X_1 und X_2 je ein Widerstand von 510 Ω nach Masse geschaltet (Bild 3-15).

Das Signal des Oszillators ist mit gleicher Frequenz, einem Tastverhältnis von 50 Prozent und versehen mit TTL-Pegel am Ausgang OSC, Pin 12 erhältlich. Intern wird die Oszillatorfrequenz durch drei geteilt und mit einem Tastverhältnis von ⅓ High-Pegel zu ⅔ Low-Pegel am Ausgang CLK, Pin 8 ausgegeben, von wo es den Prozessoren 8086 bzw. 8088 als Taktsignal zugeführt wird. Parallel dazu wird dieses Signal intern nochmals durch zwei geteilt und mit einem Tastverhältnis von 50 Prozent zum Takten von peripheren Bausteinen am Ausgang PCLK, Pin 2 ausge-

geben. Die Frequenz des PCLK-Ausgangs ist somit ⅙ der Quarzfrequenz. Die positive Flanke des CLK-Signals wird ferner zur Synchronisierung der Ready- und Reset-Signale benutzt (Bild 3-14).

Mit dem Eingang $F/\overline{C}$ Läßt sich die Herkunft des Oszillatorsignals steuern. Mit Masse an diesem Eingang ist der Quarz Signalquelle, mit High-Pegel ein externes Rechtecksignal, das am Eingang EFI (External Frequency Input) anliegt (Bild 3-16). Auch dieses Signal am Eingang EFI durchläuft den dreifachen Vorteiler für die CLK-Ausgabe und den sechsfachen für die Peripherietaktung PCLK. In diesem Falle läßt sich der Oszillatorteil unabhängig als weitere Taktquelle für andere Zwecke verwenden. Das Signal liegt ungeteilt am Ausgang OSC, Pin 12 an. Möchte man ihn nicht verwenden, sind die X_1- und X_2-Eingänge über jeweils einen 510-Ω-Widerstand an Plus zu legen. Das soll verhindern, daß sich die Eingänge eine Vorspannung anlegen, durch unkontrolliertes Schwingen HF-Rauschen erzeugen und den Stromverbrauch in die Höhe treiben.

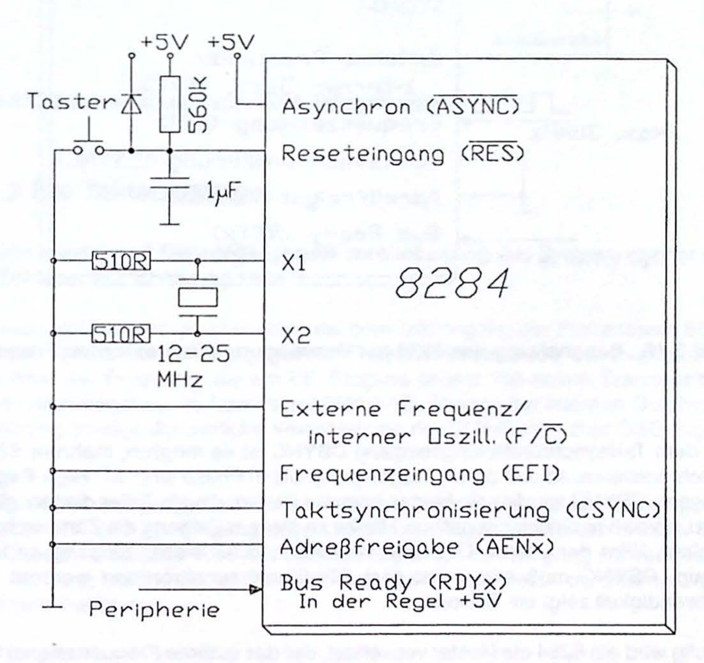


Bild 3-15. Standardbeschaltung des 8284

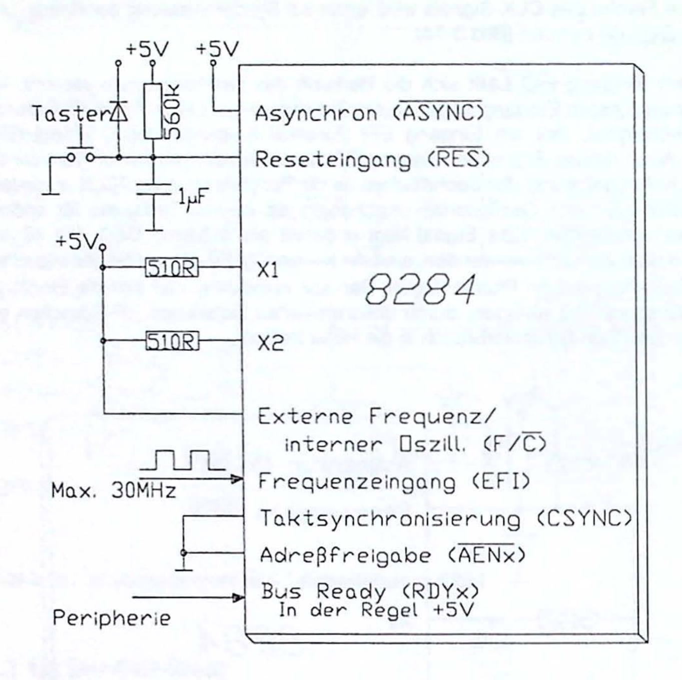


Bild 3-16. Beschaltung des 8284 bei Verwendung einer externen Frequenz

Mit dem Taktsynchronisierungseingang CSYNC ist es möglich, mehrere 8284 zu synchronisieren, so daß deren Ausgangssignale in Phase sind. Mit High-Pegel am Eingang CSYNC werden die beiden internen Zähler, die als Teiler dienen, rückgesetzt, so daß nach einer negativen Flanke an diesem Eingang die Zählung bei Null beginnt. Wird der interne Oszillator verwendet, muß dieser Eingang an Masse liegen. CSYNC muß extern mit dem EFI-Signal synchronisiert werden. Diese Notwendigkeit zeigt ein Beispiel:

Häufig wird ein 8284 als Master verwendet, der das externe Frequenzsignal für die EFI-Eingänge weiterer 8284 synchronisiert. Das wird durch die Verwendung zweier Schottky-Flip-Flops erreicht (Bild 3-17).

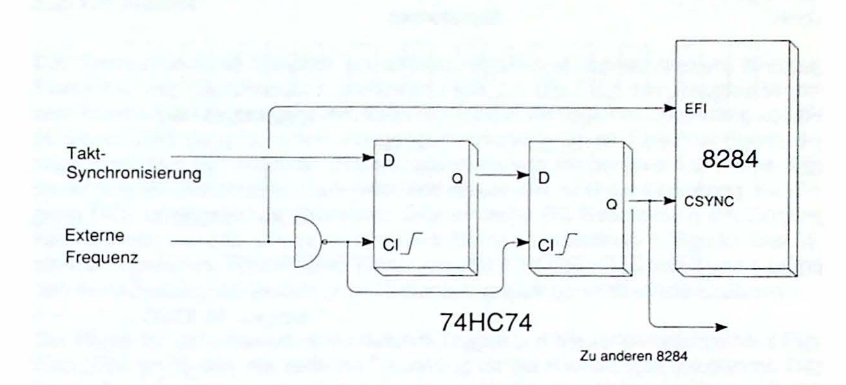


Bild 3-17. Kaskadierung von 8284-Taktgeneratoren

3.2.1.3 Die Taktausgänge

Der 8284 besitzt drei Taktausgänge, die zur Versorgung des Systems mit der nötigen Zeitbasis zur Verfügung stehen.

Der Ausgang CLK wird üblicherweise mit dem Takteingang der Prozessoren 8088 bzw. 8086 verbunden. Die ausgegebene Frequenz ist ein Drittel der Oszillatorfrequenz bzw. der Frequenz, die am EFI-Eingang anliegt, mit einem Tastverhältnis von ⅓ High-Pegel zu ⅔ Low-Pegel (Bild 3-18). Wegen der internen Durchlaufverzögerung beträgt die zeitliche Verschiebung der CLK-Flanke zum OSC-Signal 22 ns.

Das Signal des Ausgangs OSC ist ein TTL-Abbild der Frequenz des Schwingkreises an den X1- und X2- Eingängen und wird in der Regel zum Takten anderer Oszillatorbausteine benutzt. Wird ein externes Signal am EFI-Eingang verwendet, ist dieser Ausgang inaktiv, d.h. er führt High- oder Low-Pegel, wenn der interne Oszillator abgeschaltet ist.

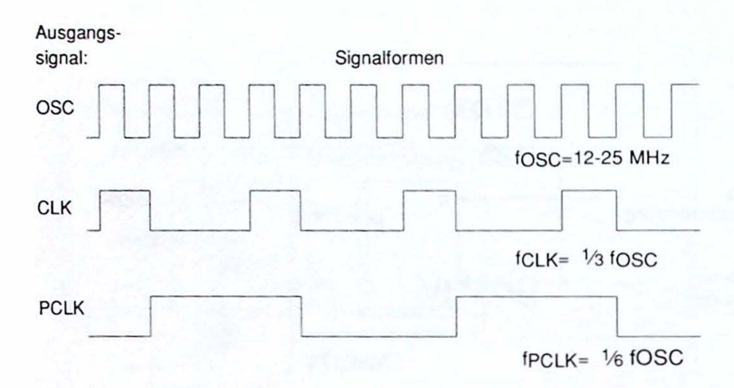


Bild 3-18. Taktausgangssignale des 8284

Der Ausgang PCLK bietet für Peripheriebausteine ein Taktsignal mit 50 Prozent Tastverhältnis an, dessen Frequenz ½ der des CLK-Ausgangs bzw. ⅙ der des OSC-Ausgangs darstellt. Wegen der internen Durchlaufverzögerung beträgt die zeitliche Verschiebung der PCLK-Flanke zum CLK-Signal 22 ns.

3.2.1.4 Reset

Der Prozessor 8088 benötigt ein aktives High-Reset-Signal, dessen minimale Pulsbreite vier CLK-Perioden umfassen muß. Da die CPU das Resetsignal mit dem internen Takt synchronisiert, kann es zu einer verzögerten Erkennung von bis zu einem Taktzyklus kommen. Ausgenommen davon ist der Einschalt-Reset, der nach Erreichen der positiven Betriebsspannung von mindestens 4,5 V eine Zeitdauer von 50 µs betragen muß. Während dieser Zeit muß die Spannung am Eingang $\overline{RES}$ unterhalb 1,05 V bleiben. Eine einfache RC-Beschaltung am Eingang kann benutzt werden, um einen Einschalt-Reset hinreichender Länge für das System zu erzeugen. Typisch sind Werte von 560 k für den Widerstand und 1 µF für den Kondensator, die jedoch ohne Bedenken größer gewählt werden können.

Der Reset-Eingang besitzt einen Schmitt-Trigger und ein synchronisierendes Flip-Flop (Bild 3-14), das die zeitliche Steuerung für die Reset-Logik übernimmt. Das Reset-Ausgangssignal ist mit der negativen Flanke des CLK-Signals synchronisiert (Bild 3-19) und kann den Haupt-Reset für das ganze Computersystem bereitstellen.

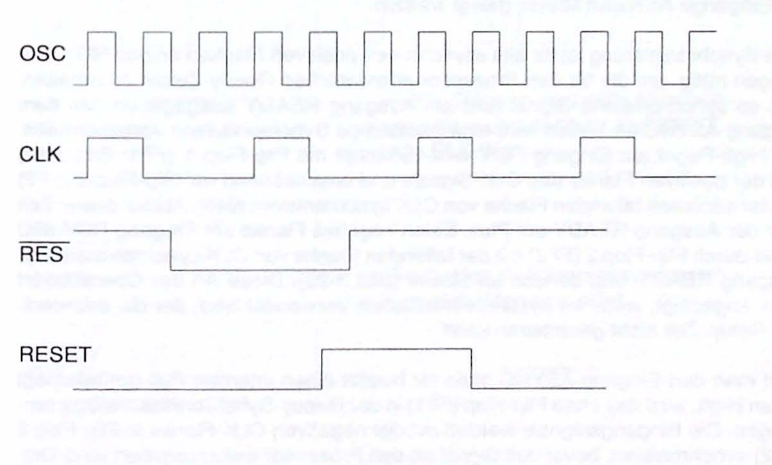


Bild 3-19. Reset-Synchronisierung im 8284

3.2.1.5 Ready-Synchronisierung

Das Ready-Signal wird in erster Linie zur Erzeugung von Wartezyklen (Wait States) genutzt, um die Verwendung von preiswerten, langsamen Speichern oder I/O-Bausteinen zu ermöglichen. Darüber hinaus wird das Signal in Multiprozessor-Systemen benötigt, um die CPU aufzufordern, vorübergehend den Zugriff auf den gemeinsamen Bus einzustellen, damit es zu keinen Konflikten beider Prozessoren kommt. Dazu wird die Ready-Synchronisierung über den 8284 benutzt.

Der 8284 kann in Systemen eingesetzt werden, die synchrone oder asynchrone Ready-Signale verwenden, indem der Eingang $\overline{ASYNC}$ auf High (synchron) oder Low (asynchron) gesetzt wird. Um den synchronen Modus zu wählen, muß der Anwender das Ready-Timing analysieren, um sicherzugehen, daß die Setup- und Haltezeiten sich im Bereich der Anforderungen der Eingänge RDY und $\overline{AEN}$ des 8284 befinden. Sollte man sich dessen nicht sicher sein, ist die asynchrone Konfiguration vorzuziehen.

Es gibt zwei Ready-Eingänge (RDY1 und RDY2), zwei Freigabeeingänge ($\overline{AEN1}$ und $\overline{AEN2}$) und einen Eingang, mit dem aus zwei Arten von Synchronisation gewählt werden kann ($\overline{ASYNC}$). Wird ein Multi-Master-System nicht benutzt, sollten die Eingänge $\overline{AENx}$ an Masse gelegt werden.

Eine Synchronisierung ist für alle asynchronen positiven Flanken an den RDY-Eingängen nötig, um die für den Prozessor erforderlichen Ready-Zeiten zu erhalten. Das so synchronisierte Signal wird am Ausgang READY ausgegeben. Mit dem Eingang $\overline{ASYNC}$ an Masse wird eine zweistufige Synchronisation vorgenommen. Ein High-Pegel am Eingang RDY wird zunächst mit Flip-Flop 1 (FF1; Bild 3-14) und der positiven Flanke des CLK-Signals und anschließend mit Flip-Flop 2 (FF2) bei der nächsten fallenden Flanke von CLK synchronisiert. Nach Ablauf dieser Zeit geht der Ausgang READY auf Plus. Einen negative Flanke am Eingang RDY wird direkt durch Flip-Flop 2 (FF2) mit der fallenden Flanke von CLK synchronisiert. Der Ausgang READY liegt danach an Masse (Bild 3-20). Diese Art der Operation ist dann angezeigt, wenn im System ein Baustein verwendet wird, der die erforderliche Setup-Zeit nicht garantieren kann.

Läßt man den Eingang $\overline{ASYNC}$ offen (er besitzt einen internen Pull-up) oder legt ihn an High, wird das erste Flip-Flop (FF1) in der Ready-Synchronisationslogik umgangen. Die Eingangssignale werden mit der negativen CLK-Flanke in Flip-Flop 2 (FF2) synchronisiert, bevor das Signal an den Prozessor weitergegeben wird. Diesen Modus wird man wählen, wenn im System Bausteine vorhanden sind, die die erforderliche Setup-Zeit zur Verfügung stellen.

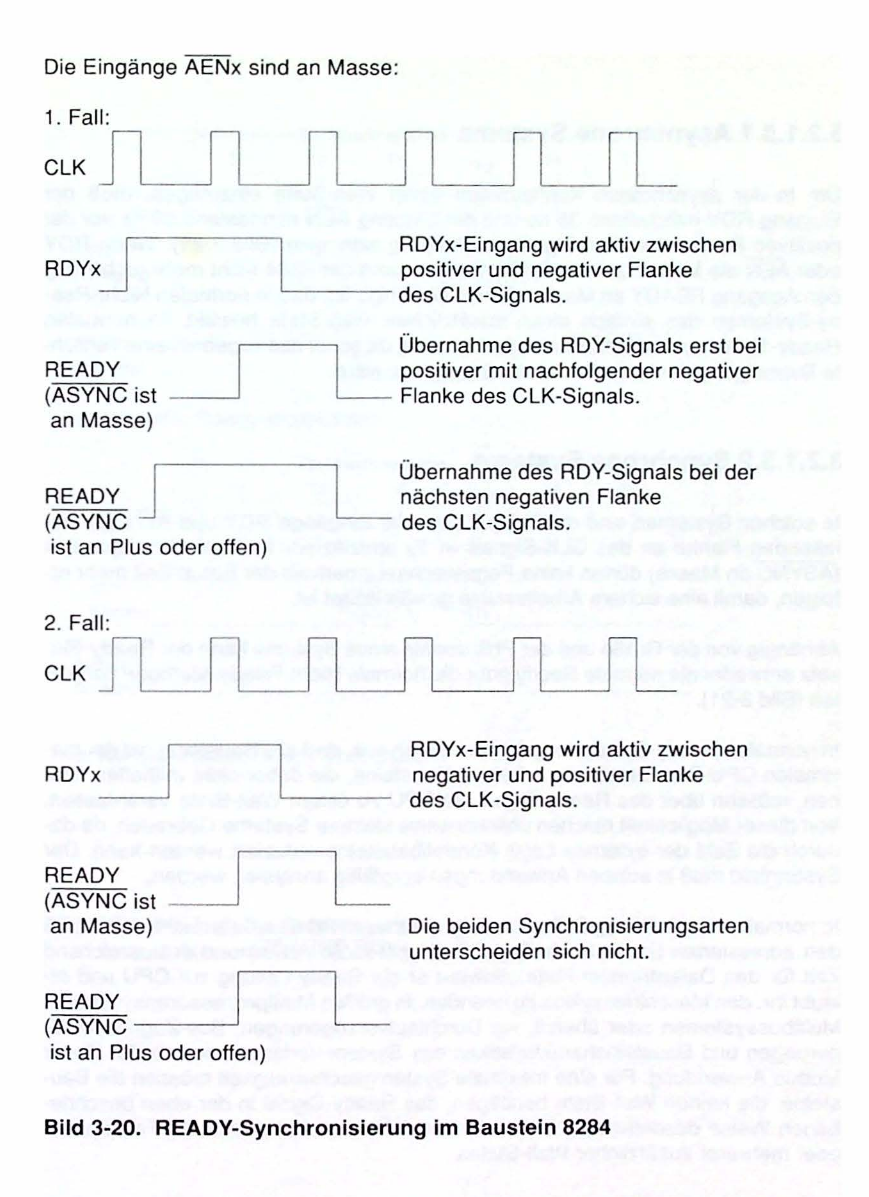


Bild 3-20. READY-Synchronisierung im Baustein 8284

3.2.1.5.1 Asynchrone Systeme

Um in der asynchronen Konfiguration einen Wait-State einzufügen, muß der Eingang RDY mindestens 35 ns und der Eingang $\overline{AEN}$ mindestens 50 ns vor der positiven Flanke des CLK-Signals in State T_2 aktiv sein (Bild 3-21). Wenn RDY oder $\overline{AEN}$ die Minimalzeiten nicht einhalten, kann der 8284 nicht mehr rechtzeitig den Ausgang READY an Masse ziehen. Die Folge ist, daß in normalen Nicht-Ready-Systemen das einfach einen zusätzlichen Wait-State bewirkt. In normalen Ready-Systemen muß das verhindert werden, da sonst das Ergebnis eine verfrühte Beendigung des aktuellen Maschinenzyklus wäre.

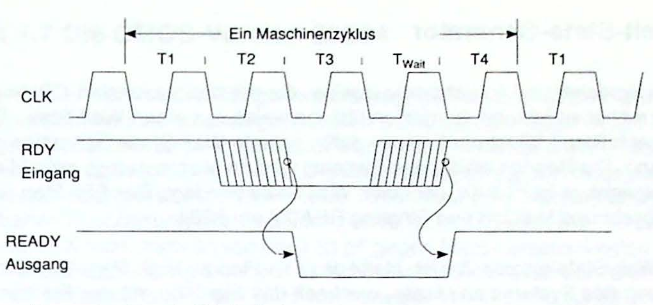
3.2.1.5.2 Synchrone Systeme

In solchen Systemen sind die Setup-Zeiten der Eingänge RDY und $\overline{AEN}$ von der fallenden Flanke an des CLK-Signals in T_2 spezifiziert. In dieser Konfiguration ($\overline{ASYNC}$ an Masse) dürfen keine Pegelwechsel innerhalb der Setup-Zeit mehr erfolgen, damit eine sichere Arbeitsweise gewährleistet ist.

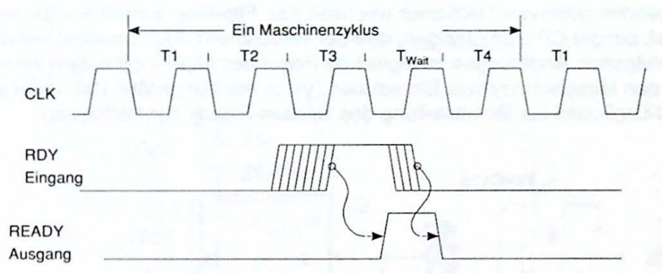
Abhängig von der Größe und der Philosophie eines Systems kann der Ready-Einsatz entweder die normale Ready oder die normale Nicht-Ready-Methode benutzten (Bild 3-21).

In normalen Ready-Systemen geht man davon aus, daß alle Bausteine mit der maximalen CPU-Bus-Bandbreite arbeiten. Bausteine, die dabei nicht mithalten können, müssen über das Ready-Signal die CPU zu einem Wait-State veranlassen. Von dieser Möglichkeit machen üblicherweise kleinere Systeme Gebrauch, da dadurch die Zahl der externen Logik-Kontrollbausteine reduziert werden kann. Der Systemtakt muß in solchen Anwendungen sorgfältig analysiert werden.

In normalen Nicht-Ready-Systemen liegt die zweite Art des Gebrauchs. Wenn auf den adressierten Baustein ein Zugriff erfolgt ($\overline{RD}/\overline{WR}/\overline{INTA}$) und er ausreichend Zeit für den Datentransfer hatte, aktiviert er die Ready-Leitung zur CPU und erlaubt ihr, den Maschinenzyklus zu beenden. In großen Multiprozessorensystemen, Multibussystemen oder überall, wo Durchlaufverzögerungen, Bus-Zugriffsverzögerungen und Bausteincharakteristiken das System verlangsamen, findet dieser Modus Anwendung. Für eine maximale Systemgeschwindigkeit müssen die Bausteine, die keinen Wait-State benötigen, das Ready-Signal in der eben beschriebenen Weise desaktivieren. Treten Fehler dabei auf, hat es nur die Folge eines oder mehrerer zusätzlicher Wait-States.



a. Normaler Ready-Wait-State



b. Normaler Nicht-Ready-Wait-State

Bild 3-21. Timing des Wait-State: a. in normalen Ready- und b. in normalen Nicht-Ready-Systemen.

3.2.1.6 Wait-State-Generator

Die meisten Speicher und Peripheriebausteine, die mit der maximalen CPU-Frequenz nicht mithalten können, benötigen üblicherweise nur einen Wait-State. Die Schaltung nach Bild 3-22 ist ein Beispiel dafür, wie ein Wait-State-Generator gebaut sein kann. Die Ready-Leitung des Systems wird an Masse gelegt, sobald ein Baustein freigegeben ist ($\overline{CS}$=0), der einen Wait-State benötigt. Das Flip-Flop wird mit ALE gelöscht und aktiviert den Eingang READY am 8284.

Wenn kein Wait-State erforderlich ist, bleibt das Flip-Flop an High-Pegel. Liegt die Ready-Leitung des Systems an Masse, wechselt das Flip-Flop mit der Taktflanke von T_2 den logischen Pegel von Masse nach Plus und bewirkt einen Wait-State. Mit der folgenden positiven Taktflanke wechselt das Flip-Flop erneut seinen logischen Pegel, um der CPU anzuzeigen, daß der Maschinenzyklus beendet werden kann. Nachfolgende Änderungen im logischen Pegel des Flip-Flops haben keinen Einfluß auf den Maschinenzyklus. Ein solcher Zyklus stellt ungefähr 100 ns für die Chip-Select-Logik und zur Bereitstellung des System-Ready zur Verfügung.

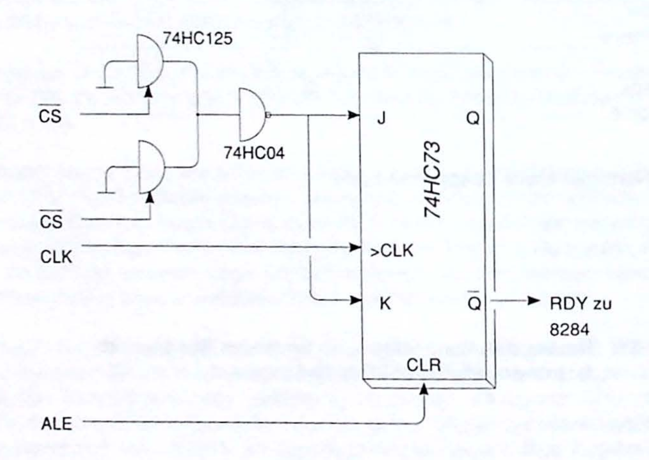


Bild 3-22. Wait-State-Generator

Die Stromaufnahme überschreitet 170 mA nicht.

3.2.1.7 Die CMOS-Version 82C84

Die CMOS-Version des Taktgenerators trägt die Nummer 82C84 und ist pin- und weitestgehend funktionskompatibel zum 8284. So kann auch er mit Frequenzen zwischen 15 und 25 MHz betrieben werden und versorgt damit Prozessoren mit einer Taktfrequenz bis zu 8 MHz. Die Stromaufnahme ist allerdings merklich reduziert, sie zeichnet sich durch maximal 10 mA aus. Ein Unterschied besteht in der externen Beschaltung des Oszillatorkreises. In ihm müssen die 510-Ω-Widerstände durch Kondensatoren von etwa 33 pF gegen Masse ersetzt werden. Zur genauen Frequenzjustierung empfiehlt es sich, einen Kondensator auf 18 pF zu reduzieren und parallel dazu einen Trimmkondensator mit 45 pF zu schalten (Bild 3-23).

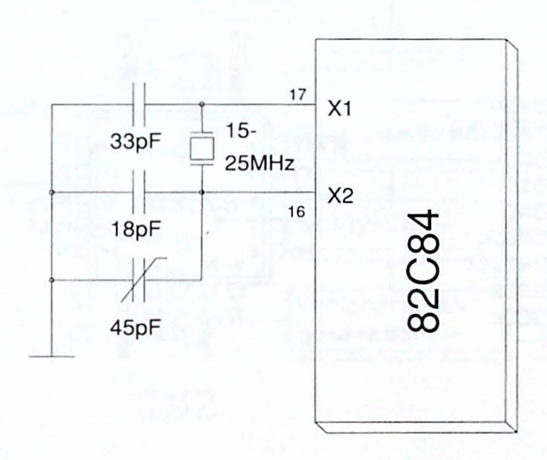


Bild 3-23. Oszillatorbeschaltung in der CMOS-Version (82C84)

3.2.2 Der Taktgenerator 82C284

3.2.2.1 Übersicht

Der 82C284 ist ein Taktgenerator für den Prozessor 80286. Die Anschlußbelegung zeigt Bild 3-24. Er ist in mancherlei Hinsicht dem 8284 ähnlich, unterscheidet sich aber von ihm wesentliche durch die Prozessortakterzeugung. Die Quarzfrequenz an den Oszillatoreingängen (X1 und X2) wird ohne Teilung, aber verstärkt an den Ausgang CLK geführt. Da der Prozessor 80286 den Takt intern durch zwei teilt, muß die gewählte Taktfrequenz das Doppelte des gewünschten Prozessortaktes sein.

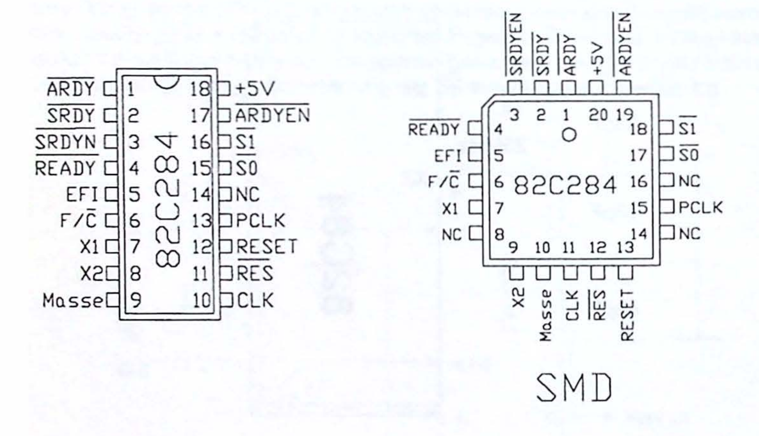


Bild 3-24. Anschlußbelegung des Taktgenerators 82C284

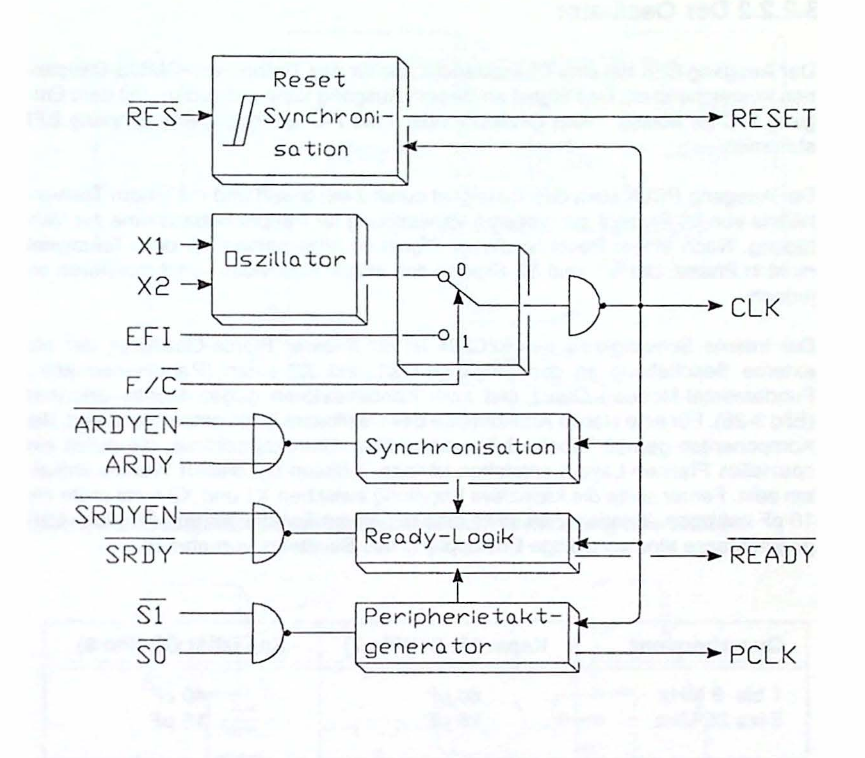


Bild 3-25. Blockdiagramm des Taktgenerators 82C284

3.2.2.2 Der Oszillator

Der Ausgang CLK hat eine Charakteristik, die für das Treiben von CMOS-Bausteinen ausreichend ist. Das Signal an diesem Ausgang kann entweder - mit dem Eingang $F/\overline{C}$ an Masse - vom Oszillator oder - mit $F/\overline{C}$ an Plus - vom Eingang EFI stammen.

Der Ausgang PCLK stellt das Taktsignal durch Zwei geteilt und mit einem Tastverhältnis von 50 Prozent zur weiteren Verwendung für Peripheriebausteine zur Verfügung. Nach einem Reset ist dieses Signal im allgemeinen mit dem Taktsignal nicht in Phase. Die $\overline{S1}$- und $\overline{S2}$-Signale des ersten Buszyklus synchronisieren es jedoch.

Der interne Schwingkreis des 82C284 ist ein linearer Pierce-Oszillator, der als externe Beschaltung an den Eingängen X1 und X2 einen (Parallelresonanz-, Fundamental-Modus-) Quarz und zwei Kondensatoren gegen Masse erfordert (Bild 3-26). Für eine stabile Arbeitsweise des Oszillators ist es empfehlenswert, die Komponenten gemäß Tabelle 3-4 auszuwählen. Streukapazitäten, die durch ein spezielles Platinen-Layout entstehen können, müssen bei diesen Werten enthalten sein. Ferner sollte die kapazitive Kopplung zwischen X1 und X2 nicht mehr als 10 pF betragen. In jedem Fall sollte man mit einem Tantal-Elko von Pin 16 (+5 V) gegen Masse eine sorgfältige Entkopplung des Bausteins vornehmen.

Quarzfrequenz	Kapazität C1 (Pin 7)	Kapazität C2 (Pin 8)
1 bis 8 MHz	60 pF	40 pF
8 bis 25 MHz	25 pF	15 pF

Tabelle 3-4. Werte für externe Oszillatorbeschaltung (82C284)

Da der Ausgang CLK über sehr kurze Anstiegs- und Abfallszeiten verfügt, ist es bei Frequenzen oberhalb 10 MHz empfehlenswert, um Signalreflexionen und Rauschen zu unterdrücken, einen Widerstand von 10 bis 74 Ω seriell in die CLK-Leitung zu legen (Bild 3-27). Das wird als serielle Termination bezeichnet. Generell sollte der Widerstandswert zusammen mit der Impedanz des Ausgangs der restlichen Leitungsimpedanz entsprechen.

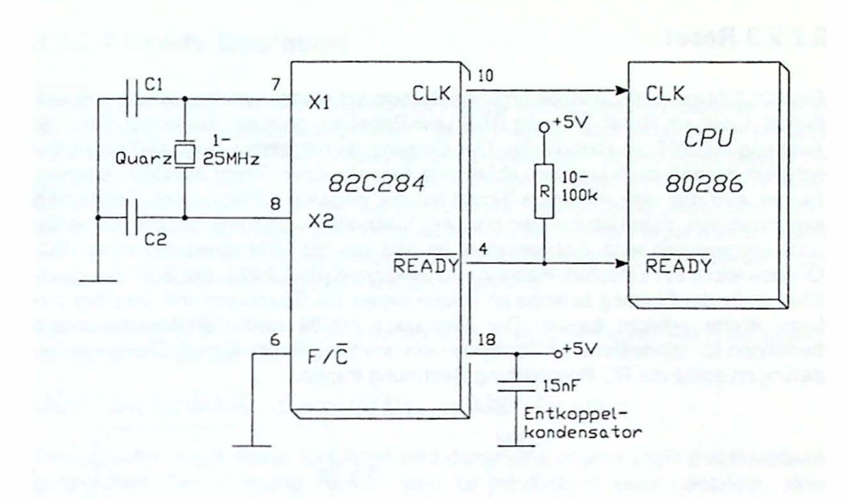


Bild 3-26. Oszillatorbeschaltung und Ready-Verbindung des 82C284

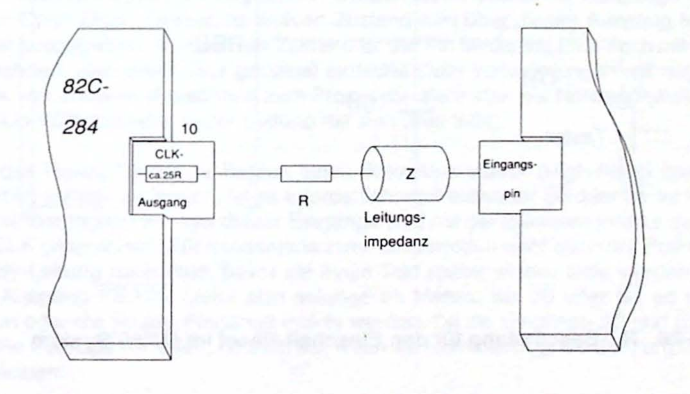


Bild 3-27. Unterdrückung von Signalreflexionen in der Taktleitung des 82C284

3.2.2.3 Reset

Der 82C284 eset I82C284versorgt das System mit einem synchronisierten $\overline{\text{Reset}}$-Signal. Liegt am Reset-Eingang $\overline{\text{RES}}$ Low-Pegel an, geht der korrespondierende Ausgang RESET an High-Pegel. Der Eingang ist mit einem Schmitt-Trigger beschaltet, so daß auch langsam abfallende Signale einen Reset auslösen können. Ferner wird das einkommende Signal mit der negativen Flanke des Taktsignals synchronisiert; dabei können ein bis zwei Taktzyklen vergehen, bis das aktive Signal ausgegeben wird. Insbesondere ist dadurch mit Hilfe eines einfachen RC-Gliedes leicht ein Einschalt-Reset zu verwirklichen (Bild 3-28). Bei Spannungsanstieg sollte der Eingang solange an Masse liegen, bis Spannung und Takt ihre stabilen Werte erreicht haben. Der Prozessor 80286 und Peripheriebausteine benötigen für mindestens 16 Taktzyklen ein positives Reset-Signal. Diesen Anforderungen sollte die RC-Beschaltung Rechnung tragen.

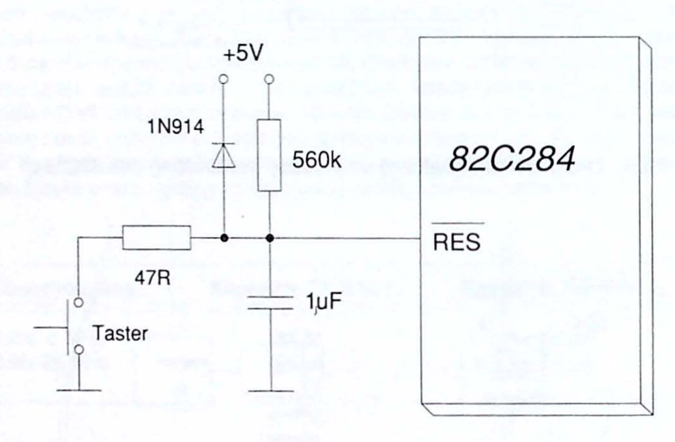


Bild 3-28. RC-Beschaltung für den Einschalt-Reset im 80286-System

3.2.2.4 Ready-Operation

Der 82C284 akzeptiert zwei Quellen von Ready-Systemsignalen, die eine laufende Busoperation beeinflussen können. Es ist möglich, eine asynchrone ($\overline{ARDY}$) oder eine synchrone ($\overline{SRDY}$) Ready-Quelle zu benutzen. Jeder dieser beiden Signaleingänge besitzt einen korrespondierenden Freigabe-Pin $\overline{ARDEN}$ bzw. $\overline{SRDYEN}$, mit dessen Hilfe man die Herkunft der Ready-Quelle auswählen kann, die sich auf den laufenden Buszyklus auswirken soll. In der Regel werden diese beiden Eingänge mit den Ausgängen eines Adreßdekoders verbunden sein, der einen von den beiden Freigabeeingängen auswählt.

Der Ausgang $\overline{READY}$ (Pin 4) wird genau dann aktiv (Low-Pegel), wenn die folgende logische Aussage erfüllt ist:

($\overline{SDRY}$ *und* $\overline{SDRYEN}$ = 0) *oder* ($\overline{ARDY}$ *und* $\overline{ARDYEN}$ = 0)

Der logische Pegel dieser Eingänge wird durch eine interne Logik des Bausteins gespeichert. Der Ausgang $\overline{READY}$ wird für mindestens zwei Taktzyklen aktiv bleiben.

Damit von anderen Quellen über dieselbe Leitung eine READY-Anforderung an den Prozessor 80286 weitergegeben werden kann, besitzt der Ausgang $\overline{READY}$ einen Open-Drain-Treiber. Im aktiven Zustand wird über diesen Ausgang Massepegel ausgegeben, im inaktiven Zustand ist der Pin weder mit Plus noch mit Minus verbunden, also offen. Das gestattet einfache *Oder*-Verknüpfungen mit negativer Logik von anderen Bausteinen zum Prozessor, zieht aber die Notwendigkeit eines Pull-Up-Widerstandes in der Leitung mit sich (Bild 3-26).

Um das Ready-Signal zu Beginn eines Buszyklus inaktiv (High-Pegel über den Pull-Up) werden zu lassen, ist es erforderlich, daß entweder $\overline{S0}$ oder $\overline{S1}$ an Masse liegen. Der logische Pegel dieser Eingänge wird mit der fallenden Flanke des Taktes CLK gespeichert. Für mindestens zwei Taktperioden zieht dann der Pull-Up die Ready-Leitung nach Plus, bevor sie einen Takt später wieder aktiv werden kann. Der Ausgang $\overline{READY}$ bleibt also solange an Masse, bis $\overline{S0}$ oder $\overline{S1}$ an Masse gehen oder die Ready-Eingänge inaktiv werden. Da die Eingänge $\overline{S0}$ und $\overline{S1}$ über interne Pull-Ups verfügen, können sie, wenn sie nicht benötigt werden, unbeschaltet bleiben.

3.2.2.5 Elektrische Eigenschaften

Die Stromaufnahme des Bausteins hängt, wie bei CMOS-Bausteinen üblich, stark von der Frequenz ab, mit der er betrieben wird. Bild 3-29 zeigt die Stromaufnahme in Abhängigkeit von der Frequenz.

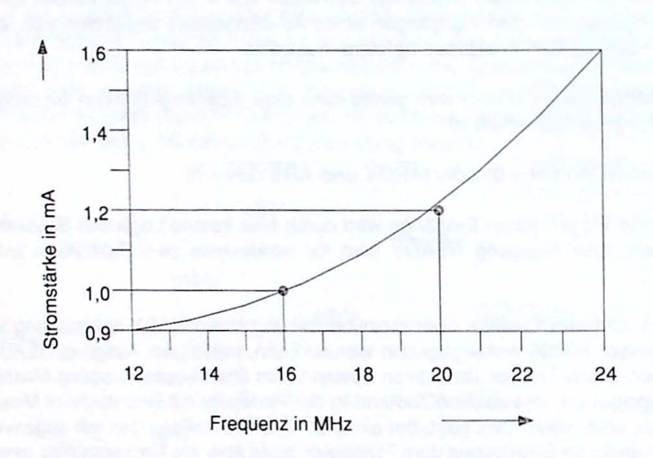


Bild 3-29. Stromaufnahme des 82C284 bei normalen Bedingungen

3.2.3 Der Taktgenerator 82384

3.2.3.1 Übersicht

Der 82384 ist in erster Linie dazu entwickelt, die Taktsignale für Systeme bereitzustellen, die den Prozessor 80386 als Zentraleinheit benutzen. Der Baustein besitzt einen Schwingkreis (X1,X2), der mit wenigen externen Elementen ergänzt werden muß, einen externen Frequenzeingang (EFI), falls auf den integrierten Schwingkreis verzichtet werden soll, eine Reset- ($\overline{\text{RES}}$) und Adreßstatussynchronisierung ($\overline{\text{ADS}}$). An den Ausgängen stellt er den Oszillatortakt (CLK2), die Hälfte der Oszillatorfrequenz (CLK), sowie das synchronisierte Reset- (RESET) und Adreßstatussignal ($\overline{\text{ADSO}}$) zur Verfügung (Bild 3-31).

Obgleich der Baustein nur zehn aktive Anschlüsse besitzt (zwölf mit den Spannungsversorgungsanschlüssen), ist er in einem 18poligen Gehäuse untergebracht (Bild 3-30). Beachtenswert ist die Lage der Spannungsversorgungsanschlüsse: +5 V muß an die Pins 1, 5 und 14, Masse an die Pins 2, 10, 11 und 17 gegeben werden. Der Anschluß 3 (NC = No Connect) sollte unbedingt frei bleiben.

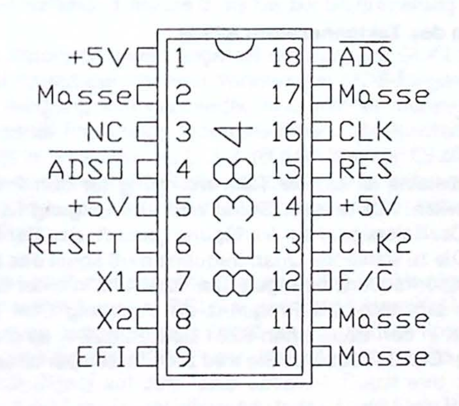


Bild 3-30. Pin-Belegung des Taktgenerators 82384

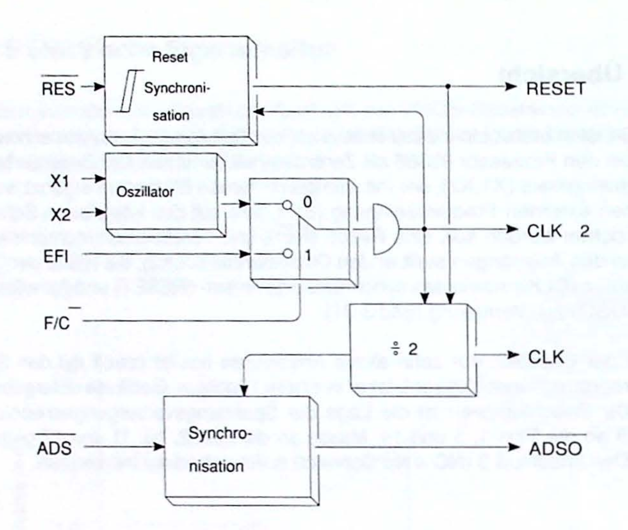


Bild 3-31. Blockdiagramm des Taktgenerators 82384

3.2.3.2 Der Oszillator

Die Hauptaufgabe des Bausteins ist es, die Taktversorgung für den Prozessor 80386 zur Verfügung zu stellen. Aus diesem Grund wird am Ausgang CLK2 das gepufferte und verstärkte Oszillatorsignal zur Verfügung gestellt, das der Prozessor intern durch Zwei teilt. Die zu wählende Quarzfrequenz muß somit das Doppelte der erforderlichen Prozessorfrequenz betragen. Der 82384 ist in einer 32-MHz- und einer 40-MHz-Version erhältlich. Die Frequenz am Ausgang CLK (er entspricht dem Ausgang PCLK in den Bausteinen 8284 bzw. 82C284) ist die Hälfte der Oszillator- und damit der CLK2-Frequenz. Sie wird zum Takten peripherer Bausteine verwendet.

Der Schwingkreis, der an den Eingängen X1 und X2 mit externen Bauteilen beschaltet werden muß, ist ein Dritter-Oberton-Oszillator. Bild 3-32 zeigt die externe Beschaltung des Oszillators. Der Auswahl des Quarzes ist besondere Aufmerk-

samkeit zu widmen, da nicht seine Fundamentalschwingung wie im 82C284, sondern der dritte Oberton zur Frequenzerzeugung herangezogen wird. Die Eigenschaften des Quarzes müssen folgende Bedingungen erfüllen:

- Leistungsaufnahme: Sie sollte 1 mW betragen.
- Maximale Shunt-Kapazität: 7 pF
- Maximaler Serienwiderstand: 40 Ω
- Resonanzverhalten: Dritter-Oberton-Parallel-Resonanz
  (Ein Dritter-Oberton-Seriell-Resonanz Quarz kann ebenfalls verwendet werden; es ist dabei aber mit einer Frequenzabweichung von 0,01% zu rechnen.)

Für eine stabile Oszillatoroperation ist die Dimensionierung des externen Schwingkreises nach Bild 3-32 empfehlenswert. Die genannten Werte sollten die Streukapazitäten der Leiterbahnen und des speziellen Layouts berücksichtigen und miteinschließen. Insbesondere sollte die kapazitive Kopplung zwischen den Eingängen X1 und X2 nicht mehr als 10 pF betragen.

Verglichen mit einem Oszillator, der die Fundamentalfrequenz des Quarzes benutzt, z.B der des 82284, benötigt ein Dritter-Oberton-Oszillator längere Zeit zum Anschwingen und für die Stabilisierung. Wenn die Zeit bei einem im Fundamentalmodus schwingenden Oszillator etwa 1 ms beträgt, kann sie bei einem Dritten-Oberton-Oszillator 1 ms bis 3 ms bis zur Stabilisierung dauern.

Das Oszillatorsignal wird ungeteilt am Ausgang CLK2 zur Verfügung gestellt und kann auf Grund der internen Verstärkung MOS-Eingänge treiben. Das Signal an diesem Ausgang kann entweder vom internen Schwingkreis oder vom Eingang EFI (External Frequency Input) stammen. Die Auswahl erfolgt mit dem Eingang $F/\overline{C}$. Liegt er an Masse, ist CLK2 mit dem internen Oszillator verbunden, liegt er an Plus, ist der Eingang EFI die Taktquelle. Das externe Frequenzsignal sollte ein Tastverhältnis von 50 Prozent aufweisen und über kurze Anstiegs- und Abfallszeiten verfügen.

Der 82384 stellt einen zweiten Taktausgang zur Verfügung: CLK. Dieser Ausgang entspricht dem Ausgang PCLK in den Bausteinen 8284/82284. Er spiegelt die halbe Frequenz des Ausgangs CKL2 wieder, verfügt über ein Tastverhältnis von 50 Prozent und besitzt MOS-Pegel. Das Blockschaltbild zeigt (Bild 3-31), daß sich das Reset-Signal auf den Teiler auswirkt. Damit wird das Signal CLK synchronisiert und führt Low-Pegel während Phase 1 und High-Pegel während Phase 2 des 80386-Buszyklus. Diese Synchronisierung braucht nur einmal am Ende eines Resets vorgenommen zu werden, da bei der weiteren Tätigkeit des Prozessors 80386 keine Phasenänderung mehr auftreten wird.

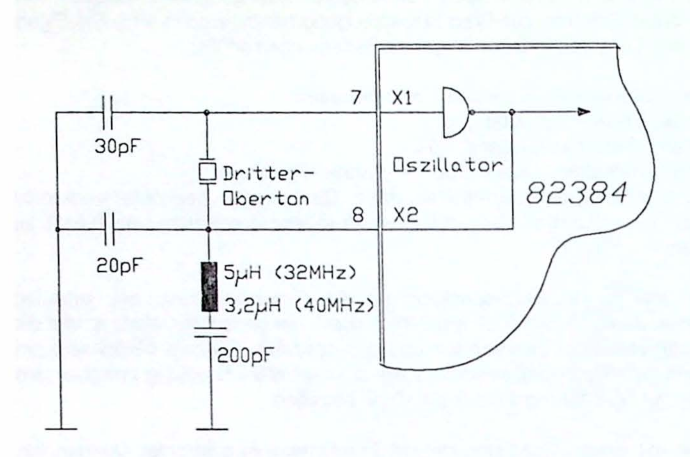


Bild 3-32. Externe Oszillatorbeschaltung des 82384.

3.2.3.3 Taktausgänge

Wie in allen elektrischen Leitungen, in denen sehr schnelle und kurze Impulse mit äußerst geringen Anstiegs- und Abfallszeiten vorkommen, tritt auch bei den beiden Taktausgängen das Problem der Signalreflexion, des Rauschens und der Übersprechung auf. Um diese negativen Effekte auf ein Minimum zu reduzieren, müssen in die Leitungen Dämpfungsglieder eingebaut werden. Das kann durch den Einbau von Widerständen in die Leitungen erreicht werden und, falls nötig, durch eine kleine kapazitive Last, die zusammen mit der anzusprechenden Pin-Kapazität mindestens den Wert von 80 pF erreicht. Dieser Kondensator ist also nur dann nötig, wenn die Summe der Leitungs- und Pin-Kapazitäten weniger als 80 pF beträgt. Bild 3-33 zeigt den Einbau des RC-Gliedes und die Besonderheiten der Strangverzweigung bei der Ansteuerung mehrerer Bausteine (IC). Pro Verzweigung sollten nicht mehr als zwei Bausteine seriell angeschlossen werden, da sonst der erste IC leicht Störungen durch Signalreflexionen und Rauschen erfahren könnte. Die ungefähren Werte für Widerstand und Kondensator finden sich in Bild 3-34.

Bei der Entwicklung des Platinen-Layouts für die Ausgänge CLK2 und CLK sollten die Widerstände unmittelbar am Baustein 82384 und der Kondensator direkt im Anschluß daran angebracht werden. Eventuelle Weiterverzweigungen sollten in der Weise angelegt sein, daß sie die gleichen Kapazitäten und die gleiche Länge besitzten.

Die HF-Abstrahlung ist bei den hohen Frequenzen ein Problem und kann durch kurze Leiterbahnen in Grenzen gehalten werden. Bei einer Zahl von vier Abzweigungen sollte die Länge einer Verzweigung 15 cm nicht übersteigen. Darüber hinaus ist es empfehlenswert, die beiden Taktleitungen möglichst zwischen den Masse- und +5 V-Stromversorgungsleitungen einzubetten.

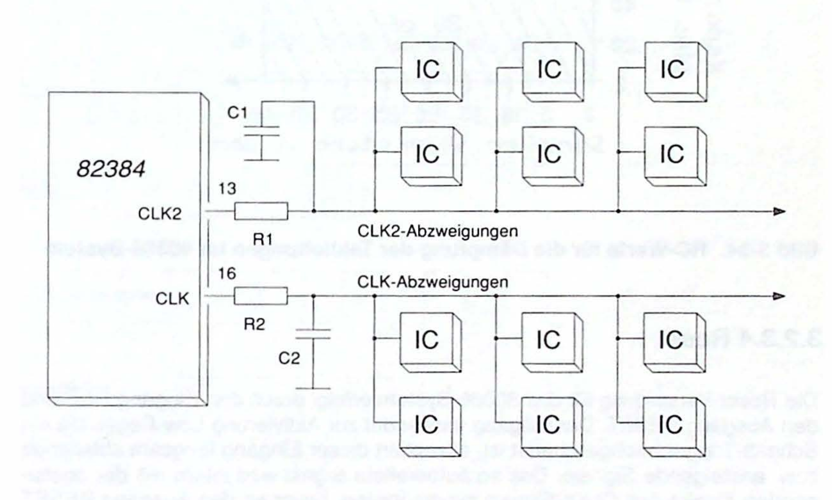


Bild 3-33. Dämpfung und Verzweigungs-Layout der Taktverbindungen im 80386-System

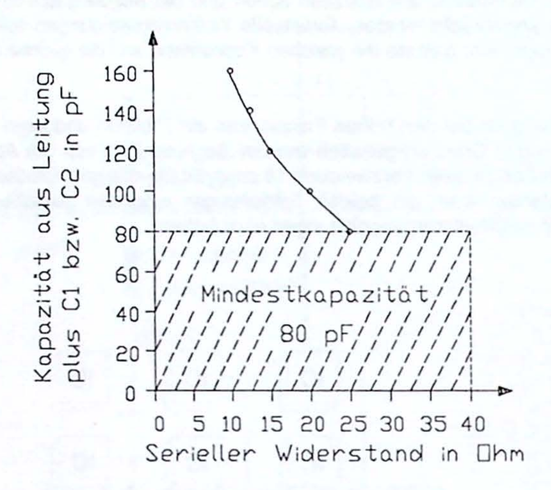


Bild 3-34. RC-Werte für die Dämpfung der Taktleitungen im 80386-System

3.2.3.4 Reset

Die Reset-Versorgung für das 80386-System erfolgt durch den Eingang $\overline{RES}$ und den Ausgang RESET. Der Eingang verwendet zur Aktivierung Low-Pegel. Da ein Schmitt-Trigger nachgeschaltet ist, akzeptiert dieser Eingang langsam abfallende bzw. ansteigende Signale. Das so aufbereitete Signal wird intern mit der ansteigenden Flanke des CLK2-Signals synchronisiert, bevor es den Ausgang RESET mit Plus-Pegel aktiviert. Somit vergehen zwischen Ein- und Ausgabe des Signals ein bis zwei CLK2-Perioden.

Die Hauptaufgabe des Eingangs ist es, mit einer RC-Beschaltung den Einschalt-Reset für das System auszulösen. Die externe Beschaltung dafür erfolgt in gleicher Weise wie für den 80C284 (Bild 3-28) mit dem Unterschied, daß der zeitbestimmende Widerstand von 10 kΩ durch einen von 20 bis 47 kΩ zu ersetzen ist.

Das hat seinen Grund darin, daß der Oszillator als Dritter-Oberton-Oszillator wesentlich länger zum Anschwingen und zu seiner Stabilisierung benötigt als ein Oszillator, der im Fundamental-Modus schwingt. Somit ist es in jedem Fall besser, die Werte für das RC-Glied eher größer als zu klein zu wählen.

Mit der Ausgabe eines aktiven Reset-Signals wird der Ausgang CLK synchronisiert, indem er auf High-Pegel gesetzt wird (Bild 3-35). Nach Inaktivierung des Eingangs $\overline{RES}$ geht der Ausgang RESET bedingt durch eine interne Logik (Bild 3-31) mit der positiven Flanke des CLK-Signals an Masse. Diese Synchronisierung benötigt der 80386 zur Übereinstimmung mit der eigenen internen Phase. Die Phase des CLK-Signals ist daher ein exakter Indikator für die interne Phase des Prozessors 80386: Low-Pegel für Phase 1, High-Pegel für Phase 2.

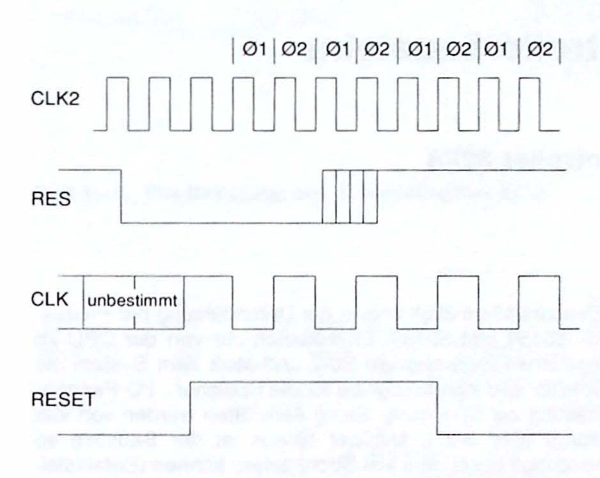


Bild 3-35. Reset-Synchronisation im 80386-System

3.2.3.5 Adreßstatus

Der Prozessor 80386 gibt jedesmal, wenn ein neuer Bus- und Adreßzyklus ausgeführt wird, ein low-aktives Statussignal $\overline{\text{ADS}}$ aus, das von Peripheriebausteinen zur Koordinierung benutzt werden kann. Eine Synchronisierung des $\overline{\text{ADS}}$-Signals kann daher für solche Schaltkreise von Nutzen sein, die ihre Taktversorgung mit dem CLK-Signal erhalten.

Die maximale Stromaufnahme des 82384 übersteigt 105 mA nicht.

3.3 Buscontrollerbausteine

3.3.1 Der Buscontroller 8288

3.3.1.1 Übersicht

Die Hauptaufgabe des Buscontrollers 8288 liegt in der Unterstützung der Prozessoren 8086, 8088, 8089, 80186 und 80188. Er dekodiert die von der CPU im Maximum-Modus ausgegebenen Statussignale $\overline{\text{S0-2}}$ und stellt dem System die daraus resultierenden Befehls- und Kontrollsignale für die Speicher-, I/O-Peripherie- und Interrupt-Handhabung zur Verfügung. Seine Aktivitäten werden von vier Kontrolleingängen gesteuert (Bild 3-37). Darüber hinaus ist der Baustein so konstruiert, daß seine Ausgänge besonders viel Strom liefern können (Befehlsleitungen 32 mA, Kontrolleitungen 16 mA), was seinen Einsatz in ausgedehnteren Systemen ratsam macht.

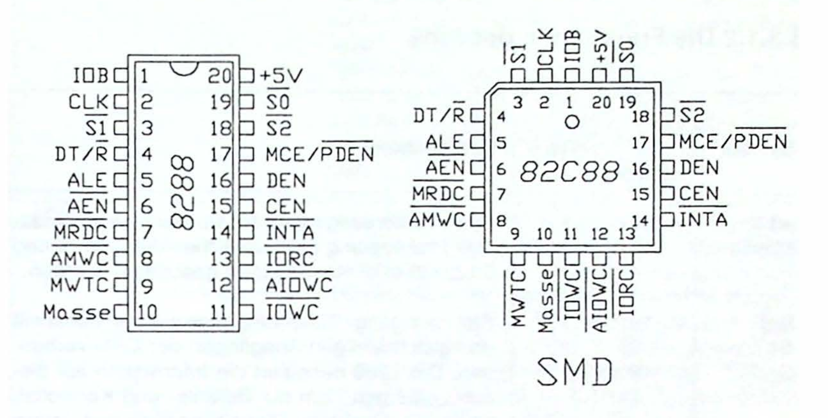


Bild 3-36. Pin-Belegung des Buscontrollers 8288

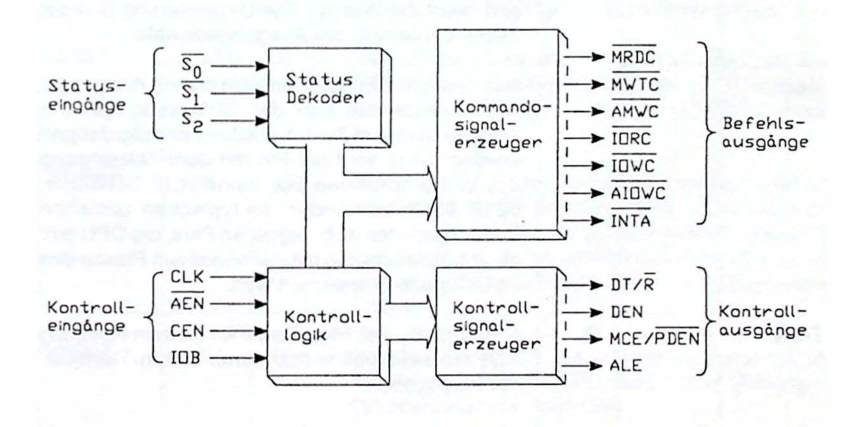


Bild 3-37. Blockschaltbild des Buscontrollers 8288

3.3.1.2 Die Funktionen der Pins

Symbol	Pin-Nummer	Typ	Beschreibung
+5 V, Masse	20, 10	I I	Stromversorgungsanschlüsse. Für eine zuverlässige Entkopplung sollte zwischen die Pins 10 und 20 ein 0,1-µF-Kondensator geschaltet werden.
S_0, S1 S_2	19, 3, 18	I I I	Statuseingang. Diese Eingänge werden direkt mit den gleichnamigen Ausgängen der CPU verbunden. Der 8288 dekodiert die Information auf diesen Leitungen , um die Befehls- und Kontrollsignale zu erzeugen. Diese Eingänge sind intern mit einem Pull-Up versehen.
CLK	2	I	Clock. Der Takteingang wird mit dem gleichnamigen Ausgang des Taktgenerators 8284 verbunden und dient der internen Synchronisierung und der Zeittaktsteuerung der Ausgangssignale.
ALE	5	O	Address Latch Enable. Mit Hilfe dieses Ausgangssignals kann die von der CPU ausgegebene Adresse in einem Zwischenspeicher aufgefangen werden. Dazu wird der Pin mit dem Takteingang des entsprechenden Bausteins (z.B. 74HC373, 8282, 8283) verbunden. Im typischen zeitlichen Verlauf geht das ALE-Signal an Plus, die CPU gibt die Adresse aus, die mit der negativen Flanke des ALE-Signals gespeichert wird.
DEN	16	O	Data Enable. Mit High-Pegel an diesem Ausgang wird ein eventuell vorhandener Daten-Transceiver freigegeben.

DT/$\overline{R}$	4	O	Data Transmit/Receive. Dieser Ausgang kontrolliert die Richtung des Datenflusses zwischen der CPU und Speicher oder I/O-Peripheriebausteine. Wenn die CPU Daten an die Peripherie schreibt, führt der Ausgang High-Pegel, und Low-Pegel beim Lesen. Der Pin wird mit dem Transmit-bzw.dem Direction-Eingang eines geeigneten Transceivers (z.B. 74HC245, 8286, 8287) verbunden.
$\overline{\text{AEN}}$	6	I	Address Enable. Mit Masse an diesem Eingang werden die Befehlsausgänge spätestens 110 ns und höchstens 250 ns nach der negativen Flanke freigegeben. Mit High-Pegel an diesem Eingang werden die Treiber der Befehlsleitungen abgeschaltet und gehen in den Tri- State. Das gilt nur, wenn der Eingang IOB an Masse liegt. Führt er High-Pegel, sind die I/OBefehlsleitungen ($\overline{\text{IORC}}$, $\overline{\text{IOWC}}$, $\overline{\text{AIOWC}}$ und $\overline{\text{INTA}}$) weiterhin aktiv.
CEN	15	I	Command Enable. Mit Masse an diesem Eingang sind alle Befehlsausgänge sowie die Signale DEN und $\overline{\text{PDEN}}$ in ihrem inaktiven Zustand (nicht Tri-State). Mit High-Pegel sind sie freigegeben.
IOB	1	I	Input/Output Bus Mode. Mit High-Pegel ist der I/O-Bus-Modus gewählt, mit Masse der Systembus-Modus. (Näheres in der Funktionsbeschreibung)
$\overline{\text{AIOWC}}$	12	O	Advanced I/O Write Command. Dieses Signal ist ein Schreibbefehl für einen I/O-Baustein, der sich vom normalen Schreibverlauf (IOWC) dadurch unterscheidet, daß er früher aktiviert wird, so daß etwas langsamere Bausteine Verwendung finden können.
$\overline{\text{IOWC}}$	11	O	I/O Write Command. Diese Leitung geht kurzzeitig an Masse, wenn die CPU Daten oder Befehle in I/O-Bausteine schreiben will.
$\overline{\text{IORC}}$	13	O	I/O Read Command. Diese Leitung geht kurz an Masse, wenn die CPU Daten aus einem I/O-Baustein lesen will.

AMWC	8	0	Advanced Memory Write Command. Dieses Signal bildet einen Schreibbefehl an einen Speicherbaustein, der sich vom normalen Schreibverlauf (MWTC) dadurch unterscheidet, daß er früher aktiviert wird. Somit können eventuell anfallende Wait-States umgangen werden.
MWTC	9	O	Memory Write Command. Diese Leitung geht beim Schreiben in einen Speicherbaustein kurz an Masse, damit der Chip die Daten übernimmt.
MRDC	7	O	Memory Read Command. Diese Leitung geht kurz an Masse, wenn die CPU Daten oder Befehle aus einem Speicher lesen möchte.
INTA	14	O	Interrupt Acknowledge. Diese Leitung wird mit dem gleichnamigen Eingang des Interruptcontrollers 8259 verbunden und dient zur Kommunikation der CPU mit dem 8259 zu Beginn der Interrupt-Routine. In der Signalform ist es mit dem IORC-Signal identisch
MCE/ PDEN	17	O	Master Cascade Enable/Peripheral Data Enable. Dieser Ausgang hat zwei Funktionen: 1. IOB an Masse: Die MCE-Funktion wird freigegeben. Das Signal wirkt während einer Interrupt-Sequenz als Freigabe für die 8259-Slaves, die nun die vom Master auf den Bus gegebene Adresse lesen, vorausgesetzt, es sind mehrere 8259 mit Master-Slave-Struktur vorhanden. 2. IOB an Plus: Die PDEN-Funktion wird freigegeben. Low-Pegel an diesem Ausgang gibt einen Daten-Transceiver frei, der nur in Verbindung mit den I/O-Befehlen (IORC, IOWC, AIOWC, INTA) die Daten zu oder von den I/O-Bausteinen fließen läßt.

Tabelle 3-5. Pin-Beschreibung des 8288

3.3.1.3 Die Betriebsarten

1. Die Dekodierung der Statusleitung:

Die Befehlslogik dekodiert die drei Statuslinien S_{0-2} des Prozessors und stellt das entsprechende Signal auf den Kontrolleitungen zur Verfügung (Tabelle 3-6).

$\overline{S2}$	$\overline{S1}$	$\overline{S0}$	**Prozessorstatus**	**Signal**
0	0	0	Interrupt-Sequenz	$\overline{INTA}$
0	0	1	Daten aus einem I/O-Baustein lesen	$\overline{IORC}$
0	1	0	Daten oder Befehle an einen I/O-Baustein schreiben	$\overline{IOWC}/\overline{AIOWC}$
0	1	1	Halt	----
1	0	0	Befehl aus dem RAM lesen	$\overline{MRDC}$
1	0	1	Daten aus dem RAM lesen	$\overline{MRDC}$
1	1	0	Daten in das RAM schreiben	$\overline{MWTC}/\overline{AMWC}$
1	1	1	Keine Aktivität	----

Tabelle 3-6. Beziehung zwischen den Statuseingängen und den Befehlsausgängen

2. Der I/O-Bus-Modus:

Wenn der Eingang IOB an High-Pegel liegt, arbeitet der 8288 im I/O-Bus-Modus. In diesem Modus können die I/O-Befehlsleitungen ($\overline{\text{IORC}}$, $\overline{\text{IOWC}}$, $\overline{\text{AIOWC}}$, $\overline{\text{INTA}}$) nicht durch ein $\overline{\text{AEN}}$-Signal gesperrt werden und sind daher immer aktiv. Erfolgt ein Zugriff des Prozessors auf einen Peripheriebaustein, werden die I/O-Befehlsleitungen zusammen mit den Leitungen $\overline{\text{PDEN}}$ und DT/$\overline{\text{R}}$ aktiviert, die den I/O-Daten-Transceiver steuern. Die I/O-Befehlsleitungen können in dieser Konfiguration nicht für die Kontrolle des RAM-Datenbus verwendet werden, da in diesem Fall die Verwaltung durch den Prozessor nicht funktionieren würde. Somit kann der Modus dazu dienen, daß der 8288 zwei externe Busleitungen steuern kann. Es wird kein Wartezyklus benötigt, wenn die CPU auf den I/O-Bus zugreifen will. Ein üblicher Speicherzugriff benötigt, bevor er ausgeführt wird, eine Freigabe durch das Bus-Ready-Signal ($\overline{\text{AEN}}$ Low-Pegel). Somit ist es vorteilhaft, den I/O-Bus-Modus immer dann zu wählen, wenn Peripheriebausteine in einem Multiprozessorsystem einem Prozessor zugewiesen sind.

3. Der Systembus-Modus:

Wenn der Eingang IOB an Masse liegt, arbeitet der 8288 im Systembus-Modus. In diesem Modus wird keine Befehlsleitung, auch nicht die I/O-Leitungen, vor Ablauf von 110 ns nach der negativen Flanke des $\overline{\text{AEN}}$-Signals aktiviert. Dieser Modus setzt voraus, daß eine Busverwaltungslogik, z.B. der Busverwalter 8289, den Buscontroller 8288 über die $\overline{\text{AEN}}$-Leitung darüber informiert, wann der Bus frei für den Zugriff ist. Dieser Modus wird benutzt, wenn nur ein einziger Bus existiert. Hier können sowohl I/O- als auch Speicherbausteine mehr als einem Prozessor zugewiesen sein.

4. Die Befehlsausgänge:

Bezüglich der normalen Schreibsignale $\overline{\text{MWTC}}$ und $\overline{\text{IOWC}}$ gehen die Advanced-Schreibsignale $\overline{\text{AMWC}}$ und $\overline{\text{AIOWC}}$ einen Takt früher an Masse und bleiben zwei Takte lang low. Gedacht sind diese Signale für die Kommunikation mit Peripheriebausteinen oder statischen RAMs, die einen breiten Schreibimpuls benötigen, so daß die CPU keinen unnötigen Wait-Zyklus ausführen muß. Mit dem Ausgang $\overline{\text{INTA}}$ wird auf einen vom Interruptcontroller (8259) ausgelösten Interrupt geantwortet. Diese Leitung muß mit dem 8259 direkt verbunden sein und ist in seiner Wellenform identisch mit dem $\overline{\text{IORC}}$-Signal. In einer Interrupt-Sequenz wird diese Leitung zweimal aktiv. Der erste Impuls dient zum Takten des 8259, der zweite zum Einlesen des Vektor-Bytes der Interrupt-Herkunft. Näheres steht in der Beschreibung des 8259.

5. Der Master-Cascade-Ausgang MCE:

Dieser Ausgang ist nur in einem System von Interesse, das mehrere Interruptcontroller in einer Master/Slave-Struktur benutzt. Häufig werden, um Verbindungsleitungen zu sparen, die Kaskadierungsleitungen der Interruptcontroller über den Adreßbus geführt. Damit es zu keiner Kollision auf dem Adreßbus kommt, dient das MCE-Signal zur Freigabe der Kaskadierungsleitungen (Bild 3-38). Es wird während eines Interrupt-Beantwortungszyklus ausgegeben, vorausgesetzt, der 8288 befindet sich mit IOB an Masse im Systembus-Modus. In jeder Interrupt-Sequenz werden zwei Beantwortungszyklen unmittelbar hintereinander ausgeführt. Wie erwähnt wird der erste Zyklus benutzt, um den 8259 zu takten. Währenddessen liegen keine Daten oder Adressen auf dem Bus. Da der Buscontroller den ersten $\overline{INTA}$ nicht vom zweiten unterscheiden kann, muß mit einer externen Logik verhindert werden, daß das MCE-Signal einen Baustein erreicht, der die Leitungen CAS_{0-2} des 8259 mit dem lokalen Bus verbindet (z.B. drei NAND-Gatter). Mit dem Beginn des zweiten $\overline{INTA}$-Zyklus bewirkt das MCE-Signal die Aktivierung des Bausteins und die Kaskadierungsleitungen des Masters sind über den lokalen Adreßbus mit den Slaves verbunden. Die vom Master ausgegebene Adresse kann mittels ALE-Signal in einem geeigneten Speicher (z.B.74HC373) festgehalten werden. Bei der negativen Flanke des zweiten $\overline{INTA}$-Impulses gibt der angesprochene Slave den Interrupt-Vektor auf den Datenbus, wo er von der CPU gelesen wird.

6. Address-Latch-Enable ALE und Halt:

Mit der negativen Flanke des ALE-Signals wird die Adresse in einen Zwischenspeicher übernommen, da der Prozessor nachfolgend auf denselben Leitungen Daten an das RAM sendet oder von dort empfängt. Die Ausgänge der Zwischenspeicher sind mit den Adreßeingängen der RAMs verbunden. Als Adreßzwischenspeicher kommen die Bausteine 8282, 8283 oder 74HC373 in Frage. Das ALE-Signal erscheint bei jedem Maschinenzyklus und kann ferner zur Speicherung der Statusleitungen $\overline{S0-2}$ während der Halt-State-Dekodierung verwendet werden.

7. Command-Enable CEN:

Dieser Eingang entspricht den Chip-Select-Eingängen $\overline{CS}$ der I/O- oder RAM-Bausteine. Mit seiner Hilfe kann einer unter mehreren Buscontrollern zur Aktivität veranlaßt werden. Ist dieser Eingang high, funktioniert der Baustein normal, ist er low, bleiben alle Befehlsausgänge inaktiv (kein Tri-State). Das kann von Vorteil sein, wenn auf parallele Speicherbänke umgeschaltet werden soll oder verschiedene Prozessoren die Buskontrolle übernehmen wollen.

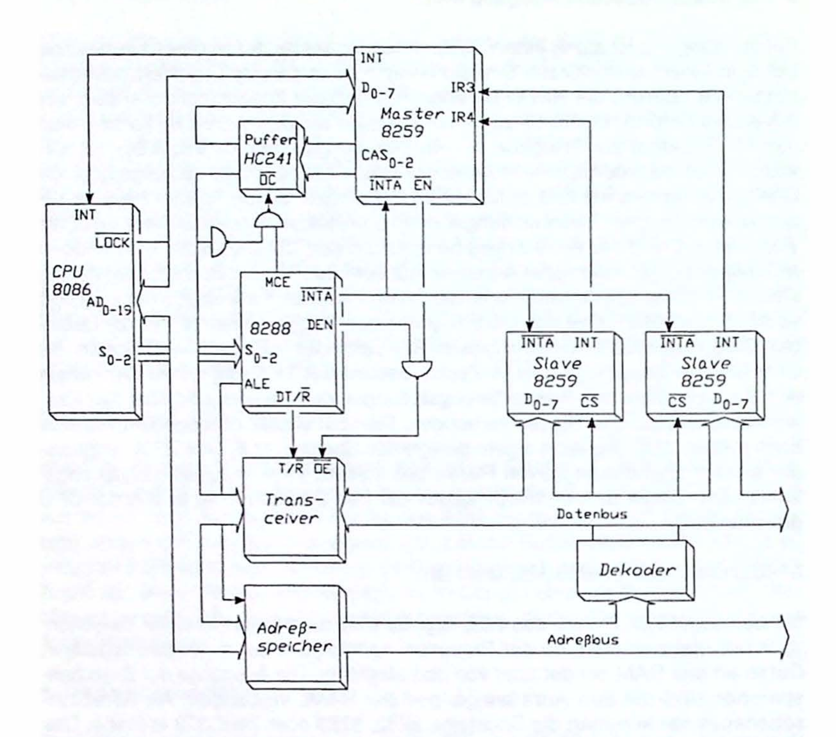


Bild 3-38. Das MCE-Signal zur Kontrolle einer 8259-Master/Slave-Struktur

3.3.1.4 Der Buscontroller in einem 8086-System

Das Bild 3-39 zeigt den Buscontroller in einem 8086-System, in dem der I/O-Bus nicht vom Speicherbus getrennt ist. Daher ist für den 8288 der Systembus-Modus gewählt (IOB an Masse). Wenn der Busverwalter 8289 fehlt, ist der Eingang $\overline{AEN}$ an Masse zu legen.

In einem 8086-System werden zwei 8Bit-Datentransceiver benötigt, da der Prozessor extern 16-Bit-Daten parallel verarbeitet. Diesem Zweck dient ferner das $\overline{BHE}$-Signal, mit dessen Hilfe 8-Bit-Speicherbausteine freigegeben werden, die das High-Byte der Daten aufnehmen. Verwendet man neuere Bausteine, die selbst eine 16-Bit-Datenbreite aufweisen, wird das $\overline{BHE}$-Signal nicht verwendet.

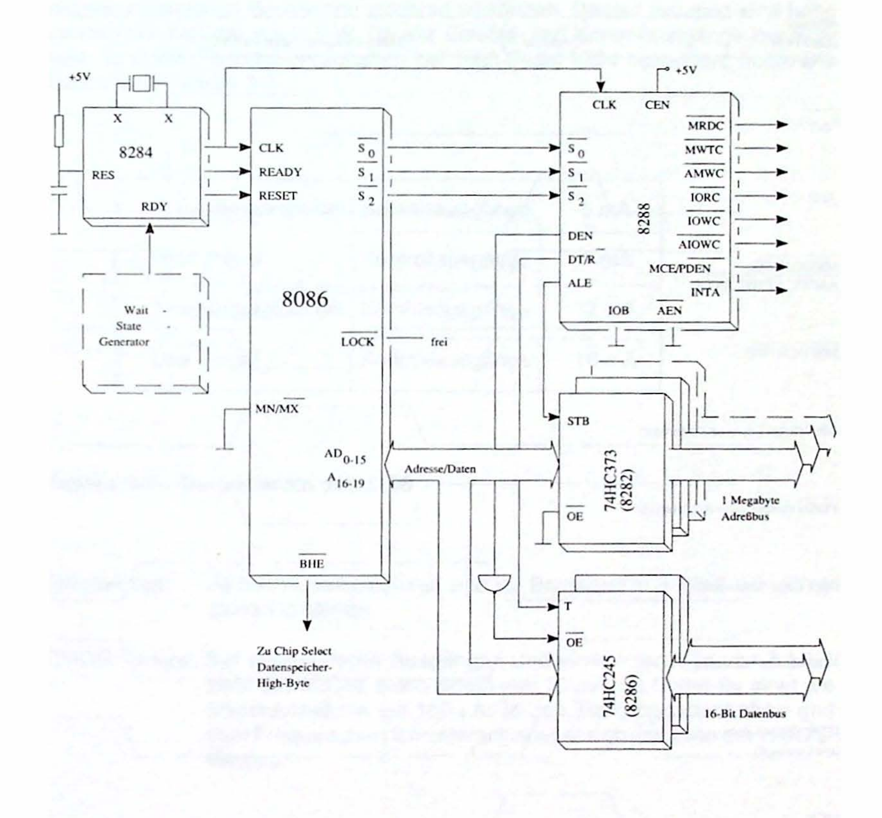


Bild 3-39. Die typische Verwendung des 8288 in einem 8086-System

Signalformen:

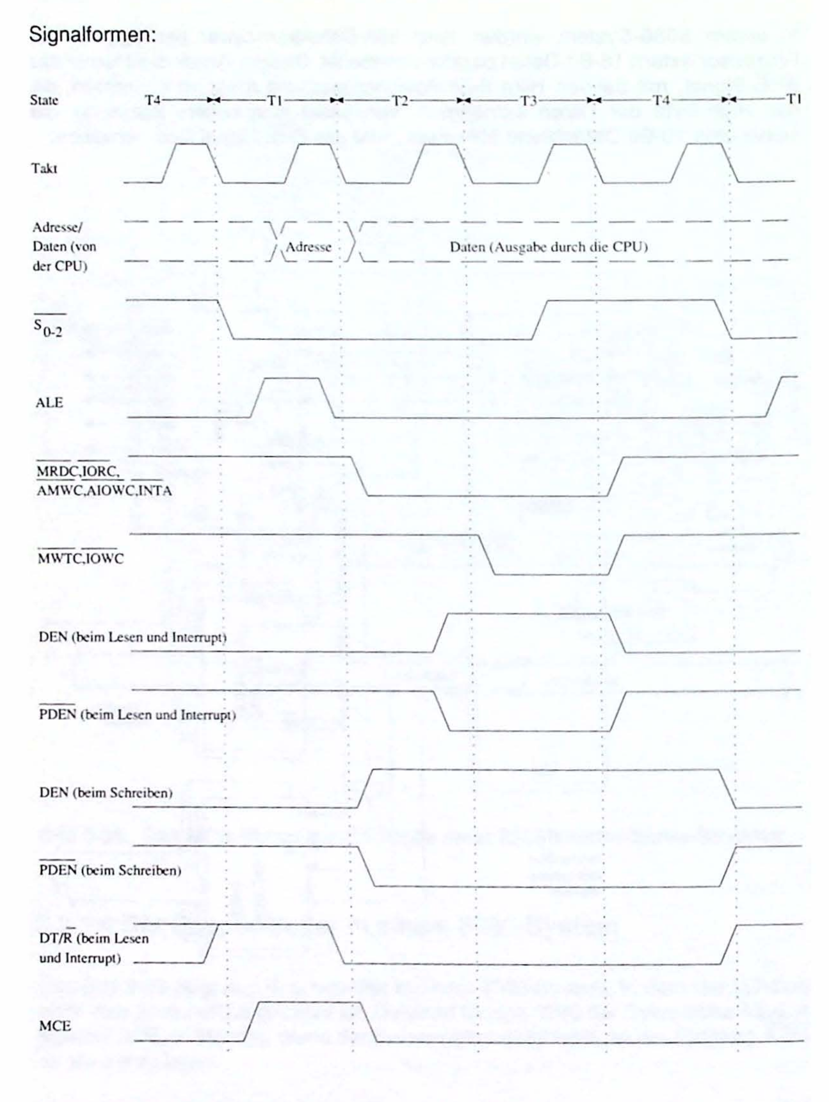


Bild 3-40. Signalformen des Buscontrollers 8288

3.3.1.5 Elektrische Eigenschaften

Der 8288 dekodiert nicht nur die Statusausgänge der CPU im Maximum-Modus, sondern liefert darüber hinaus besonders viel Strom, um die Befehls- und Kontrollsignale zahlreichen Bausteinen zuführen zu können. Daraus resultiert eine hohe Leistungsaufnahme von 1,5 W. Da alle Befehls- und Kontrollausgänge low-aktiv sind, sind die Treibereigenschaften bei High-Pegel nicht besonders hoch. Die Daten zeigt Tabelle 3-7.

Ausgangsstrom bei High-Pegel	Befehlsausgänge	-5 mA
	Kontrollausgänge	-1 mA
Ausgangsstrom bei Low-Pegel	Befehlsausgänge	32 mA
	Kontrollausgänge	16 mA

Tabelle 3-7. Treiberstrom des 8288

Frequenzen: Je nach Qualitätsstufen sind die Bausteine in 8- bis 10MHz-Versionen erhältlich.

CMOS-Version: Bei unbelasteten Ausgängen und einer Frequenz von 5 MHz zieht der 82C88 einen Strom von 10 mA. Im Stand-By sinkt die Stromaufnahme auf 100 µA. In den Treibereigenschaften und dem Frequenzbereich unterscheidet er sich nicht von der NMOS-Version.

3.3.2 Der Buscontroller 82288

3.3.2.1 Übersicht

Der 82288 dient als Buscontroller in einem 80286er-Prozessorsystem. Seine Aufgabe ist die Kontrolle der Adreßspeicher und der Datentransceiver, sowie die Lieferung des notwendigen Treiberstroms. Ferner stellt er dem System standardisierte Befehlsausgänge zur Verfügung. Sie sind synchronisiert und erfüllen in jeder Hinsicht die IEEE-796-Anforderungen für den Multibus. Mit dem Eingang MB (Multibus) ist dieser Modus wählbar. Die Signalformen der Befehlsleitungen können über den Eingang Command-Delay CMDLY in gewissen Grenzen verschoben werden, um das Timing den individuellen Wünschen anzupassen.

Eine Verbindung zu mehreren Bussen wird über den Eingang CENL (Befehlsfreigabespeicher) hergestellt. Ein Adreßdekoder wählt nun einen unter mehreren 82288 aus. Der Pegel an diesem Eingang wird beim folgenden Taktzyklus gespeichert, so daß der Adreßdekoder dem Pipline-Timing des 80286 zur Verfügung steht.

Ein Bus-Sharing durch mehrere Buscontroller wird unterstützt. Der Eingang $\overline{AEN}$ verhindert, daß der Buscontroller Befehls- oder Kontrollsignale auf den Bus gibt, wenn ein anderer gerade Zugriff darauf nimmt. Gesteuert wird dieser Eingang durch den Busverwalter 82289.

Getrennte Ausgänge für die Datenfreigabe DEN und Richtung des Datenflusses DT/$\overline{R}$ überwachen die Transceiver für alle Busse. Datenkonflikte auf dem Bus werden dadurch eliminiert, daß das Transceiver-Freigabesignal DEN vor einer Änderung des Richtungssignals DT/$\overline{R}$ ausgeschaltet wird.

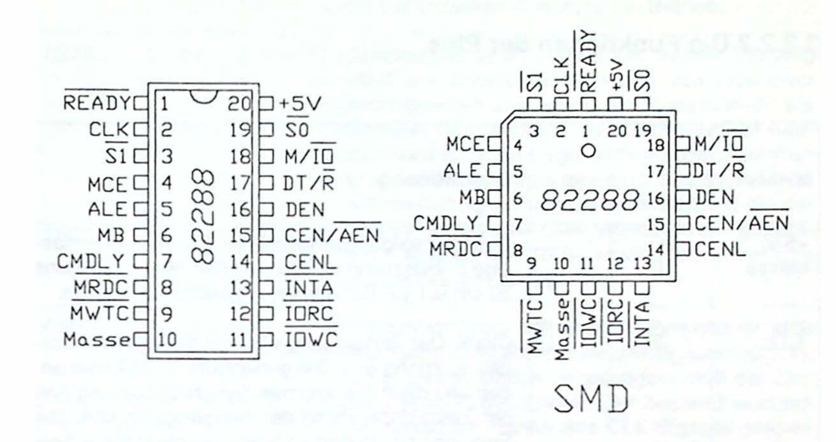


Bild 3-41. Pin-Belegung des Buscontrollers 82288

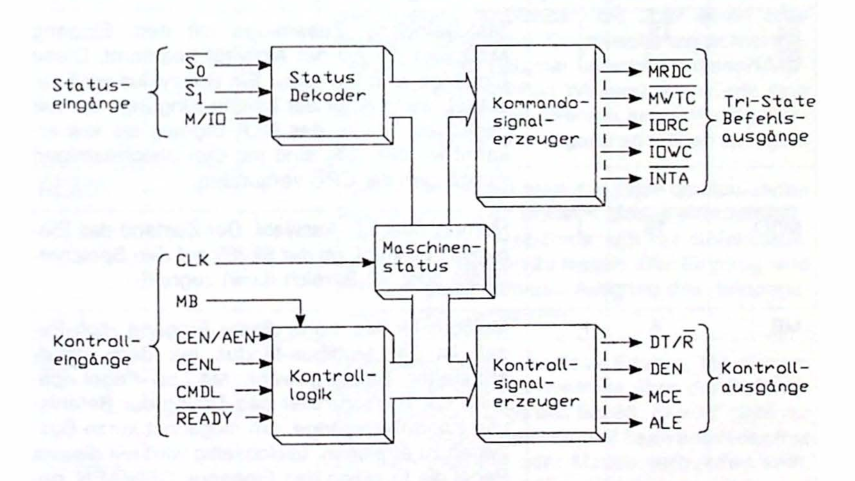


Bild 3-42. Blockdiagramm des Buscontrollers 82288

3.3.2.2 Die Funktionen der Pins

Symbol	Pin-Nummer	Typ	Beschreibung
+5 V, Masse	20, 10	I I	Stromversorgungsanschlüsse. Für eine zuverlässige Entkopplung sollte zwischen die Pins 10 und 20 ein 0,1-µF-Kondensator geschaltet werden.
CLK	2	I	Clock. Der Takteingang wird mit dem gleichnamigen Ausgang des Taktgenerators 82284 verbunden und dient der internen Synchronisierung und der Zeittaktsteuerung der Ausgangssignale. Die Frequenz ist doppelt so hoch wie die interne Prozessortaktfrequenz. Mit der fallenden Flanke werden die Pegel der Eingänge gespeichert und die der Kontrollausgänge geändert.
$\overline{S_0}$, $\overline{S_1}$	19, 3	I I	Statuseingang. Zusammen mit dem Eingang M/$\overline{IO}$ wird die Art der Aktivität bestimmt. Diese Eingänge sind low-aktiv. Ein Buszyklus wird gestartet, wenn einer der beiden Eingänge bei der negativen Flanke des CLK-Signals als low erkannt werden. Sie sind mit den gleichnamigen Ausgängen der CPU verbunden.
M/$\overline{IO}$	18	I	Memory oder I/O-Auswahl. Der Zustand des Eingangs bestimmt, ob der 82288 auf den Speicher- (High) oder I/O-Bereich (Low) zugreift.
MB	6	I	Multibus-Modus. Führt dieser Eingang High-Pegel, ist der Multibus-Modus mit dem standardisierten Timing gewählt. Mit Low-Pegel optimiert der Buscontroller das Timing der Befehls- und Kontrollausgänge, um möglichst kurze Buszyklen zu erreichen. Gleichzeitig wird mit diesem Pegel die Funktion des Eingangs CEN/$\overline{AEN}$ gewählt. Der Pegel des Eingangs wird üblicherwei-

			se hardware-mäßig festgelegt und unterliegt keiner dynamischen Änderung bei Betrieb.
CENL	14	I	Command Enable Latched. Mit diesem Eingang wählt ein Adreßdekoder einen von mehreren Buscontrollern aus, wenn ein System mehr als nur einen Bus benutzen kann. Der gewählte Baustein wird mit High-Pegel aktiviert, die restlichen werden mit Low-Pegel gesperrt. Das Signal muß nicht beständig anliegen,sondern wird mit der negativen Flanke des CLK-Signals intern gespeichert. Wird nur ein 82288 verwendet, ist der Eingang an +5 Volt zu legen.
CMDLY	7	I	Command Delay. Mit diesem Eingang ist eine verzögerte Ausgabe der Befehlssignale möglich. Um in diesen Modus zu gelangen, muß der Eingang High-Pegel führen. Der Zustand wird mit der negativen Flanke des CLK-Signals gespeichert.Solange allerdings der High-Pegel anliegt, wird kein Befehlssignal ausgegeben. Das geschieht erst, wenn die negative Flanke des CLK-Signal Low-Pegel erkennt. Somit ist eine Verzögerung erreichbar, die über einen oder mehrere Takte reicht. Ein zwischenzeitlich eingehendes $\overline{\text{READY}}$-Signal unterbricht diesen Vorgang mit der gleichen Wirkung, als würde kein Befehl ausgegeben. Werden keine Verzögerungen benötigt, ist der Eingang an Masse zu legen.
$\overline{\text{READY}}$	1	I	Ready. Das Signal zeigt das Ende des laufenden Buszyklus an. Der Multibus-Modus erfordert mindestens einen Wait-State, um die Befehlsausgänge aktiv werden zu lassen. Der Eingang wird mit dem gleichnamigen Ausgang des Taktgenerators 82284 verbunden.
CEN/$\overline{\text{AEN}}$	15	I	Command Enable/Address Enable. Mit diesem Eingang erfolgt die Kontrolle über die Befehls- und DEN-Ausgänge des 82288. Er wird nicht mit dem CLK-Signal getaktet und kann entweder fest mit dem Plus-Pol oder Masse verbunden sein. Wird der Eingang MB mit Plus verbunden, hat Pin 15 $\overline{\text{AEN}}$-Funktion. Masse am $\overline{\text{AEN}}$-Eingang

			zeigt an, daß die CPU dem 82288 die Kontrolle über den gemeinsamen Bus eingeräumt hat. Die Befehlsausgänge des Buscontrollers können den Tri-State nun verlassen und werden inaktiv (High). Plus am $\overline{\text{AEN}}$-Eingang zeigt an, daß die CPU den gemeinsamen Bus nicht kontrolliert und überführt die Befehlsausgänge in den Tri-State, sowie den Ausgang DEN an Masse. In der Regel wird $\overline{\text{AEN}}$ durch den Busverwalter 82289 kontrolliert, der genau dann das Signal $\overline{\text{AEN}}$ aktiviert, wenn der Verwalter auf den dem Buscontroller zugehörigen Bus zugreift. Wird der Eingang MB mit Masse verbunden, hat Pin 15 CEN-Funktion. High-Pegel erlaubt eine Aktivierung der Befehls- und DEN-Ausgänge, Low-Pegel inaktiviert sie, versetzt sie aber nicht in den Tri-State.
ALE	5	O	Address Latch Enable. Mit der negative Flanke des Signals wird die auf dem Bus anstehende Adresse in geeignete Adreßzwischenspeicher aufgefangen. Bei einem Halt-Befehl wird das ALE-Signal nicht ausgegeben, es kann darüberhinaus durch keinen Kontrolleingang beeinflußt werden.
MCE	4	O	Master Cascade Enable. Wie im 8288 wird dieser Ausgang nur dann verwendet, wenn im System mehrere Interruptcontroller 8259 mit Master/Slave-Struktur vorhanden sind. Das Signal ist nur während des Interrupt-Antwortzyklus aktiv und wird zur Freigabe eines Puffers verwendet, der die Slave-Adresse auf den System-Adreßbus legt. Mittels ALE-Signal müssen diese drei Adreßleitungen für die Slaves gespeichert werden.
DEN	16	O	Data Enable. Der Ausgang kontrolliert die Daten-Transceiver für den Datenbus. Im Multibus-Modus wird DEN beim Schreiben verzögert ausgegeben.
DT/$\overline{\text{R}}$	17	O	Data Transmit/Receive. Der Ausgang bestimmt die Richtung des Datenflusses auf dem Datenbus. Mit High-Pegel gehen die Daten von der

			CPU zur Peripherie, mit Low-Pegel von dort zur CPU. Bei Pegeländerungen an diesem Ausgang wird immer kurzzeitig das DEN-Signal inaktiviert, damit es zu keinen Buskonflikten kommt. Das DT/$\overline{R}$-Signal kann nicht durch die Kontrolleingänge beeinflußt werden.
$\overline{IOWC}$	11	O	I/O Write Command. Schreibbefehlsleitung an einen I/O-Baustein. Der Zeitpunkt für die Ausgabe des Signals wird durch die Eingänge MB und CMDLY bestimmt.
$\overline{IORC}$	12	O	I/O Read Command. Lesebefehlsleitung an einen I/O-Baustein. Der Zeitpunkt für die Ausgabe des Signals wird durch die Leitungen MB und CMDLY bestimmt.
$\overline{MWTC}$	9	O	Memory Write Command. Schreibbefehlsleitung an einen Speicher. Der Zeitpunkt für die Ausgabe des Signals wird durch die Leitungen MB und CMDLY bestimmt.
$\overline{MRDC}$	8	O	Memory Read Command. Lesebefehlsleitung für einen Speicher. Der Zeitpunkt für die Ausgabe des Signals wird durch die Leitungen MB und CMDLY bestimmt.
$\overline{INTA}$	13	O	Interrupt Acknowledge. Der Ausgang dient als Antwortleitung auf einen vom Interruptcontroller 8259 angemeldeten Interrupt. Das Signal wird zweimal hintereinander ausgegeben (Abschnitt: Interruptcontroller 8259). Der Zeitpunkt für die Ausgabe des Signals wird durch die Leitungen MB und CMDLY bestimmt.

Tabelle 3-8. Pin-Beschreibung des Buscontrollers 82288

3.3.2.3 Die Betriebsarten

Von dem Controllerbaustein 82288 können zwei Arten von Bussen versorgt werden: Multibus- und Nicht-Multibus-Systeme. Wenn der Eingang MB an High-Pegel liegt, ist das Multibus-Timing selektiert. In diesem Modus verzögert der 82288 die Ausgabe der Befehls- und Datensignale, um den Anforderungen des IEEE-796-Standards gerecht zu werden. Wegen dieser Verzögerung ist mindestens ein Wait-State erforderlich. Im Nicht-Multibus-Modus wird die Verzögerung nicht wirksam und erfordert daher keine Wait-States. Der Eingang MB tangiert lediglich den DEN- und die Befehlsausgänge.

M/$\overline{IO}$	$\overline{S1}$	$\overline{S0}$	Signal	DT/$\overline{R}$	ALE,DEN-Signale?	MCE-Signal?
0	0	0	$\overline{INTA}$	0	Ja	Ja
0	0	1	$\overline{IORC}$	0	Ja	Nein
0	1	0	$\overline{IOWC}$	1	Ja	Nein
0	1	1	-----	1	Nein	Nein
1	0	0	Halt: ---	1	Nein	Nein
1	0	1	$\overline{MRDC}$	0	Ja	Nein
1	1	0	$\overline{MWTC}$	1	Ja	Nein
1	1	1	-----	1	Nein	Nein

Tabelle 3-9. Dekodierung der Statusleitungen durch den Buscontroller 82288

3.3.2.4 Befehls- und Kontrollausgänge

Die Art der vom 82288 ausgeführten Aktivität wird durch die Zustände der Steuerleitungen M/$\overline{IO}$, $\overline{S1}$, und $\overline{S2}$ bestimmt. Tabelle 3-9 zeigt die dekodierten Aktivitäten, ferner die Zustände der Ausgänge DT/$\overline{R}$, ALE, DEN und MCE. Buszyklen treten in drei Arten auf: Lesen, Schreiben und Halt. Ein Lesezugriff kann sich auf das Lesen

des Speichers (Befehle oder Daten), das Lesen der Peripherie (I/O-Bausteine) oder das Lesen des Interruptcontrollers 8259 ($\overline{INTA}$-Signal) beziehen. Die Signalformen für die entsprechenden Befehls-, Kontrollausgänge und Steuereingänge sind für alle Lesezugriffe identisch. Der Unterschied liegt nur darin, welcher Befehlsausgang gerade aktiviert wird. Das MCE-Signal wird nur bei der Interrupt-Antwort benötigt, und auch nur dann, wenn mehrere Interruptcontroller mit Master/Slave-Struktur vorhanden sind.

Ähnlich sind die Signalformen beim Schreiben. Sie unterscheiden sich zwar von denen der Lesesignalen, sind aber getrennt nach Befehls- und Steuerausgängen gleich.

Während des Halt-Zyklusses werden weder Befehls- noch Kontrollsignale aktiviert. Alle Steuersignale werden ignoriert, bis der nächste Buszyklus mittels $\overline{S0}$ und $\overline{S1}$ gestartet wird.

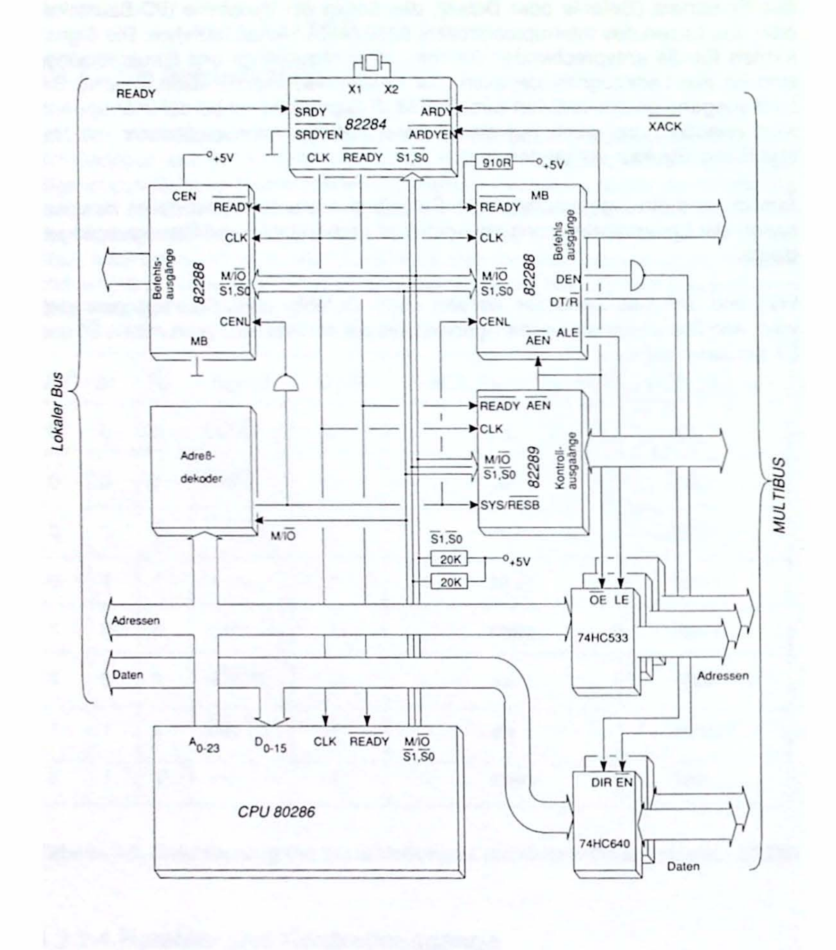


Bild 3-43. Der Buscontroller 82288 in einem 80286-System

3.3.2.5 Kontrolleingänge

Mit den Kontrolleingängen können die grundlegenden Signalformen der Befehlsausgänge beeinflußt werden. In zahlreichen 80286-Systemen hat die CPU mehr als nur einen Bus, auf den sie zugreifen kann. Üblicherweise hat die CPU zur selben Zeit nur einen Buscontroller aktiv. Auf diese Busse können auch mehrere Prozessoren Zugriff haben, allerdings immer nur einer zur gleichen Zeit.

Systeme mit mehreren Bussen benutzen zwei Eingänge des Buscontrollers zur Steuerung, es sind das der CENL- und $\overline{AEN}$-Eingang (Bild 3-43). CENL gestattet dem Buscontroller den momentanen Buszyklus zu kontrollieren, der $\overline{AEN}$-Eingang hingegen hindert den Buscontroller, seine Ausgänge aktiv werden zu lassen. $\overline{AEN}$ high bedeutet, daß ein anderer Buscontroller den gemeinsamen Bus treibt.

In Bild 3-43 sind zwei Busse zu sehen: ein lokaler Bus und ein Multibus. Nur ein einziger Bus ist zu einer bestimmten Zeit freigegeben. Mit dem Eingang CENL wird der aktive Buscontroller über den Adreßdekoder freigegeben. Der 82288, der mit dem gemeinsamen Multibus verbunden ist, muß mit CENL ausgewählt und mit $\overline{AEN}$ zum Zugriff auf den Multibus freigegeben sein, bevor er seine Steuersignale ausgeben kann.

Wie bereits erwähnt, kontrolliert der Eingang MB die Signalformen des DEN- und der Befehlsausgänge. Im Multibus-Modus werden diese Signale automatisch verzögert, so daß drei Bedingungen erfüllt sind:

1. 50 ns Setup-Time mindestens werden für die Stabilisierung der ausgegebenen Adresse garantiert, bevor die Befehlsausgänge aktiv werden.
2. 50 ns werden mindestens für das stabile Anliegen zu schreibender Daten auf dem Bus reserviert.
3. 65 ns stehen maximal zwischen Lesebefehl und Abschalten der Datentransceiver zur Verfügung.

Drei Signale werden im Multibus-Modus (MB=1) verzögert:

1. Die negativen Flanken der Lesebefehlsleitungen ($\overline{IORC}$, $\overline{MRDC}$ und $\overline{INTA}$) werden um eine CLK-Periode verzögert.

2. Die negativen Flanken der Schreibbefehlsleitungen ($\overline{IOWC}$ und $\overline{MWTC}$) werden um zwei Taktperioden verzögert.

3. Die negative Flanke des DEN-Signals für Schreibzyklen wird um eine Taktperiode verzögert.

Einige Speicher- oder Peripheriebausteine machen es erforderlich, daß die Adressen oder zu schreibenden Daten länger anliegen, als es die Standard-Signalformen vorsehen. Mit Hilfe des Eingangs CMDLY kann die Ausgabe der Befehlssignal verzögert werden. Die Kontrollausgänge ALE, MCE, DEN und DT/$\overline{R}$ werden dadurch nicht berührt. Zur Aktivierung der Befehlssignale muß das Eingangssignal zunächst als high und nachfolgend als low erkannt werden. Somit ist die Verzögerungszeit frei wählbar. Erscheint zwischendurch ein aktives $\overline{READY}$-Signal, wird die zuvor gespeicherte High-Information gelöscht und der gerade laufende Buszyklus abgebrochen.

3.3.2.6 Elektrische Eigenschaften

In seiner Eigenschaft ist der 82288 dem 8288 sehr ähnlich. Wie dieser kann er über die Befehlsausgänge 32 mA und über die Kontrollausgänge 16 mA gegen Masse liefern. Die Leistungsaufnahme beträgt 1 W. Der 82288 ist in den Frequenzbereichen von 6 bis 20 MHz erhältlich. Alle Bausteine erfordern eine große Flankensteilheit des Taktsignals von mindestens 8 ns.

3.3.3 Der Busverwalter 8289

3.3.3.1 Übersicht

Der 8289 ist ausschließlich für eine enge Anwendung konstruiert; er soll die Verwaltung des Systembusses mit den Prozessoren 8086, 8088 und 8089 übernehmen. Bei einem Zugriff von einem der genannten Prozessoren auf den gemeinsamen Bus schließt der 8289 die Aktivitäten der restlichen Prozessoren aus, indem er nur dem ersten Prozessor diese Möglichkeit einräumt. Er erzeugt die erforderlichen Signale für die Verwaltung des Multibus. Die Verwaltung kann auf vier Anforderungs- und Busübergabearten erfolgen. Da der Baustein nur dann erforderlich ist, wenn in einem System mehrere Prozessoren arbeiten, erübrigt sich seine Anwesenheit in einem Single-Prozessor-System.

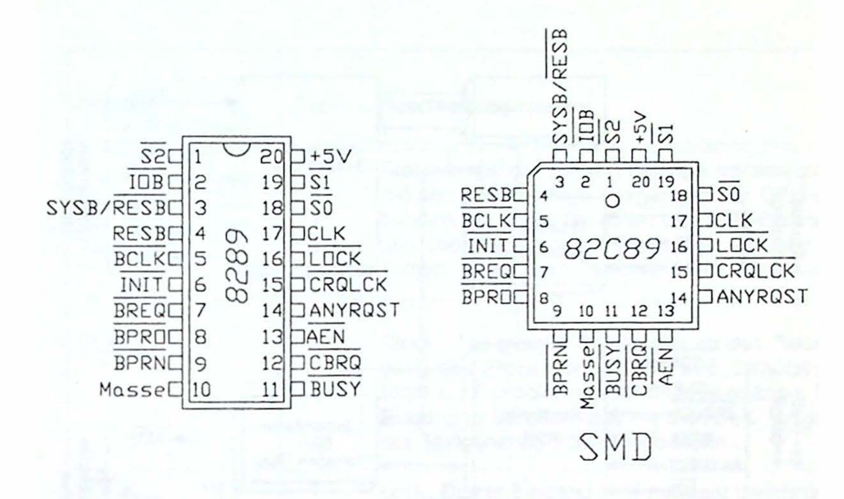


Bild 3-44. Pin-Belegung des Busverwalters 8289

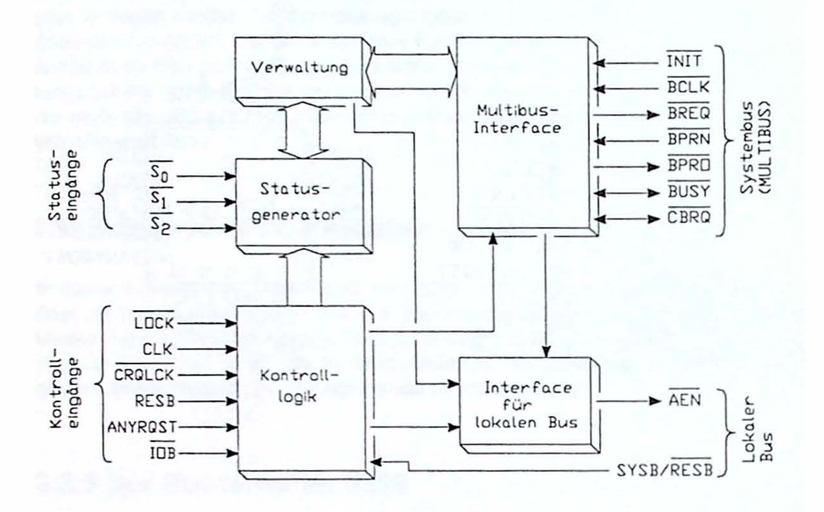


Bild 3-45. Blockdiagramm des Busverwalters 8289

3.3.3.2 Die Funktionen der Pins

Symbol	Pin-Nummer	Typ	Beschreibung
S_0, S_1, S_2	18, 19, 1	I I I	Statuseingänge. Diese Eingänge werden direkt mit den gleichnamige Ausgängen der CPU verbunden. Der 8289 dekodiert das 3-Bit-Binärwort und übergibt auf dieser Grundlage den gemeinsamen Bus dem anfordernden Prozessor.
CLK	17	I	Clock. Das gleiche Signal, das an den Takteingang des Prozessors geführt wird, benutzt der 8289 zur Dekodierung der Statuseingänge. Der Eingang ist üblicherweise mit dem CLK-Ausgang des Taktgenerators 8284 verbunden.
$\overline{\text{LOCK}}$	16	I	Lock. Dieser Eingang wird mit dem gleichnamigen Ausgang des Prozessors verbunden. Liegt Masse an diesem Pin, vergibt der 8289 die Busprivilegien unter keinen Umständen an einen Prozessor. (Beispiel: Interrupt Antwortzyklus)
$\overline{\text{CRQLCK}}$	15	I	Common Bus Request Lock Input. Das ist das Ausschlußsignal für die Verwaltung einer üblichen Busanforderung. Führt der Eingang Masse, ignoriert der 8289 die Busübergabebedingung durch das Signal CBRQ.
RESB	4	I	Resident Bus Mode Controll. Mit diesem Eingang ist der Modus des 8289 wählbar. Mit High-Pegel ist der Baustein im residenten Bus-Modus.
$\overline{\text{IOB}}$	2	I	I/O Bus Mode. Mit $\overline{\text{IOB}}$ an Masse ist der 8289 im I/O-Bus-Modus, mit High-Pegel im Memory-Bus-Modus.

ANYRQST	14	I	Any Request. Der Eingang kontrolliert die Busübergabebedingungen für den 8289. Liegt der Eingang an Masse, erfolgt die Busfreigabe nach den Bedingungen laut Tabelle 3-11. Mit High-Pegel wird der Bus übergeben, sobald der Eingang $\overline{CBRQ}$ an Masse geht. Deswegen kann ein einziger Zugriff durch einen Prozessor die Busübergabe bewirken, wenn der Eingang ANYRQST High-Pegel und der Eingang $\overline{CBRQ}$ Low-Pegel führt.
SYSB/ $\overline{RESB}$	3	I	System Bus/Resident Bus Selection. Der Eingang ist nur dann aktiv, wenn sich der 8289 im residenten Bus-Modus befindet. Masse an diesem Eingang bedeutet, daß der Prozessor nur auf seinen eigenen Bus zugreift und keinen Zugang zum Systembus haben möchte. High-Pegel zeigt dem 8289 an, daß der Prozessor den Zugang zum Systembus wünscht. Der 8289 gibt das entsprechende Anforderungssignal aus.
$\overline{BCLK}$	5	I	Bus Clock. Werden mehrere Systemplatinen kombiniert, so kann mit einem Signal an diesem Eingang die Synchronisation zur Verwaltung der anderen Platinen erreicht werden. Die Ausgangssignale des 8289 werden dann in Übereinstimmung mit diesem Taktsignal ausgegeben. Der Eingang wird üblicherweise mit der $\overline{BCLK}$-Leitung des Systembusses verbunden.
$\overline{INIT}$	6	I	Initialize. Mit Masse an diesem Eingang werden alle Busverwalter an einem Multi-Master-Bus rückgesetzt, d.h. auf ihren Anfangszustand gebracht, in dem keiner von ihnen Zugriff auf den Bus nimmt. Das Signal wird der $\overline{INIT}$-Leitung des Systembusses entnommen.
$\overline{BPRN}$	9	I	Bus Priority In. Über diesen Eingang erfährt der 8289, ob ein anderer Busverwalter höherer Priorität Zugriff auf den Bus nimmt. Mit Masse ist das nicht der Fall und der 8289 hat Zugang zum Systembus, High-Pegel sperrt den 8289. Der Eingang wird für den ersten kaskadierten Baustein

			mit Masse, für alle nachfolgen den Bausteine mit dem Ausgang $\overline{\text{BPRO}}$ des vorausgehenden Bausteins verbunden.
$\overline{\text{BPRO}}$	8	O	Bus Priority Out. Dieses Ausgangssignal zeigt mit Low- Pegel an, daß eine Busanforderung vom eigenen Baustein oder über den Eingang $\overline{\text{BPRN}}$ von einem übergeordneten 8289 erfolgte. Das Signal wird nur dann benötigt, wenn mehrere Bausteine seriell kaskadiert sind. Bei der Auswahl der Bausteine durch parallele Prioritätsdeko dierung erübrigt sich die Verwendung des Ausgangs.
$\overline{\text{BREQ}}$	7	O	Bus Request. Das Ausgangssignal ist nur dann notwendig, wenn mehrere 8289-Bausteine in einer parallelenPrioritätsauflösung betrieben werden. Die $\overline{\text{BREQ}}$-Ausgänge sämtlicher Bausteine werden den Eingängen eines Prioritäts-Enkoders (z.B.74HC148) zugeführt, welcher den so generierten Binärkode einem Dekoder zuführt, der die Prioritätsreihenfolge der Bausteine festlegt. Masse zeigt an, daß der betreffende Baustein die Buskontrolle übernehmen will.
$\overline{\text{BUSY}}$	11	I/O	Busy. Über dieses Signal wird dem System angezeigt, daß ein Zugriff auf den Systembus erfolgte. Der gerade aktive Busverwalter setzt diesen Ausgang an Masse und zeigt den anderen Bausteinen über denselben Pin an, der nun bei ihnen Eingang ist, daß er auf den Bus zugreift. Diese I/O-Funktion des Pins ist deswegen möglich, da der Ausgangspuffer einen offenen Kollektor besitzt. Die Busy-Leitung ist daher mit einem externen Pull-Up zu versehen.
$\overline{\text{CBRQ}}$	12	I/O	Common Bus Request. Als Eingang wird mit Low-Pegel über diese Leitung einem Busverwalter angezeigt, ob weniger priorisierte Bausteine den Zugriff auf den Bus haben möchten. Da der Ausgangspuffer des Pins über einen offenen Kollektor verfügt, kann er gleichzeitig Ausgabe- als auch Eingabefunktionen wahrnehmen. Die Pins aller vorhandenen Busverwalter sind miteinan der zu verbinden.

$\overline{AEN}$	13	O	Address Enable. Mit diesem Signal informiert der 8289 den Buscontroller 8288, den Oszillator 8284 und den Adreßpuffer, daß er gerade den Systembus verwaltet.

Tabelle 3-10. Pin-Beschreibung des Busverwalters 8289

3.3.3.3 Die Betriebsarten

Der 8289 arbeitet in Verbindung mit dem Buscontroller 8288 um die Prozessoren 8086, 8088 (im Maximum-Modus) und 8089 in einem Multi-Master-System zu unterstützen. Für die Erstellung von Software ist die Anwesenheit des Busverwalters 8289 unerheblich, da der Prozessor von seiner Existenz kein Kenntnis nimmt. Von der Seite des Prozessors sieht es so aus, als hätte nur er allein den Zugriff auf den Bus. Wenn der Prozessor durch äußere Gegebenheiten den Bus nicht benutzen kann, schaltet der Busverwalter 8289 den Buscontroller 8288, den Daten-Transceiver und den Adreßzwischenspeicher ab, d.h. er versetzt ihre Ausgänge in den Tri-State. Da nun der Buscontroller keine weiteren Befehlssignale mehr ausgibt, erscheint der Systembus als nicht bereit (Not Ready) und der Prozessor ist im Wait-State . Erst wenn dem Busverwalter 8289 erneut die Benutzung des Multi-Master-Busses gestattet ist, verläßt der Prozessor den Wait-State.

Der 8289 dekodiert die Statuseingänge $S_{0\text{-}2}$, die mit den gleichnamigen Prozessorausgängen verbunden sind, und meldet Anspruch auf den Systembus an oder übernimmt ihn, falls er frei ist. Die Bedingungen dafür sind in Tabelle 3-11 aufgeführt. Die folgenden vier Modi mit den entsprechenden Hardware-Layouts sind möglich:

1. Einfacher Bus-Modus:
   In diesem Modus existieren auf der Platine des betrachteten Prozessors weder Speicherbereiche noch I/O-Peripheriebausteine und der Prozessor greift ausschließlich auf den Systembus zu.

2. I/O-Bus-Modus:
   In diesem Modus gibt es nur I/O-Peripheriebausteine auf der Platine, auf die der Prozessor direkt zugreift. Der Busverwalter 8289 hat somit die Aufgabe, nur bei Zugriffe des Prozessors auf den Speicher für den Daten- und Befehlstransfer den Systembus zu bemühen.

3. Residenter Bus-Modus:
   In diesem Modus existieren sowohl I/O-Peripherie als auch der Speicher auf der Prozessorplatine und der Prozessor kann auf beides auch über den Systembus zugreifen. In diesem Fall wird das Chip Select-Signal $\overline{CS}$ dem Eingang SYSB/$\overline{RESB}$ zugeführt. Das hat zur Folge, daß der 8289 den Systembus nicht bemüht, wenn auf die Bereiche der eigenen Platine zugegriffen wird.

4. I/O-Bus-Modus, residenter Bus-Modus:
   Auch in diesem Fall sind sowohl der I/O-Bereich als auch der Speicher auf der Prozessorplatine vorhanden, es wird aber nur auf den I/O-Bereich der Platine zugegriffen. Das Chip-Select-Signal $\overline{CS}$ für den Speicher auf der Platine wird dem Eingang SYSB/$\overline{RESB}$ zugeführt. Somit wird nur dann ein Anforderungssignal für den Systembus ausgegeben, wenn auf den Systemspeicher zugegriffen wird.

Status		I/O-Bus-Modus	Residenter Bus-Modus		I/O-Bus-Modus residenter Bus-Modus		Single-Bus-Modus
		$\overline{IOB}$=0 RESB=0	$\overline{IOB}$=1; RESB=1 SYSB/$\overline{RESB}$ =		$\overline{IOB}$=0; RESB=1 SYSB/$\overline{RESB}$ =		$\overline{IOB}$ = 1 RESB = 0
Befehl	$S_{2\text{-}0}$		1	0	1	0	
Interrupt-Antwort	000	X	O	X	X	X	O
I/O-lesen	001	X	O	X	X	X	O
I/O-schreiben	010	X	O	X	X	X	O
Halt	011	X	X	X	X	X	X
Befehl einlesen	100	O	O	X	O	X	O
Speicher lesen	101	O	O	X	O	X	O
Speicher schreiben	110	O	O	X	O	X	O
Keine Aktion	111	X	X	X	X	X	X

O = An den Systembus wird ein Anforderungssignal ausgegeben

X = Der Multi-Master-Systembus kann übernommen werden

Tabelle 3-11. Busanforderungs- und Übergabebedingungen

Modus	Eingang		Zugriffsarten auf den Bus	Busübergabebedingungen
	$\overline{IOB}$	RESB		
Einfacher Bus-Modus	1	0	Alle Bus-Zugriffsarten	HLT+(TI-CBRQ)+HPBRQ
I/O-Bus-Modus	1	1	(SYSB/$\overline{RESB}$=1) ■ (Bus-Zugriffsart)	((SYSB/RESB=L+TI) ■ CBRQ) +HLT+HPBRQ
Residenter Bus-Modus	0	0	Alle Speicher-Zugriffsarten	(I/O-Zugriff+TI) ■ CBRQ +HLT+HPBRQ
I/O-Bus-Modus, residenter Bus-Modus	0	1	(SYSB/$\overline{RESB}$=1) ■ (Speicher-Zugriffsart)	((I/O-Zugriff+(SYSB/$\overline{RESB}$=0))■ CBRQ+HPBRQ+HLT

Legende:

1. Ist die Leitung $\overline{LOCK}$ an Masse, wird der Bus dem 8289 nicht überlassen.
   Ist $\overline{CRQLCK}$=0, wird der Bus auch dann nicht überlassen, wenn ihn Verwalter niedrigerer Priorität anfordern.
2. HLT Halt-Status
   TI Keine Aktion
   CBRQ $\overline{CBRQ}$ = 0
   HPBRQ Zeigt an, daß ein Verwalter höherer Priorität den Bus anfordert (BPRN = 1).
3. ■ bedeutet logische UND-Verknüpfung
   \+ bedeutet logische ODER-Verknüpfung

Tabelle 3-12. Bedingungen für die verschiedenen Bus-Modi des 8289

3.3.3.4 Multi-Prozessor-Systeme mit dem Busverwalter 8289

Beanspruchen mehrere Prozessoren denselben Systembus, so muß logischerweise jeder Prozessor zur Dekodierung seiner Statusleitungen seinen 8289-Baustein besitzen. Damit es zu keiner Konfusion auf dem Bus kommt, muß zwischen den Busverwaltern eine hierarchische Struktur bestehen, in der die Bausteine nach Prioritäten geordnet sind.

Die priorisierteren Master können erst dann auf den Bus zugreifen, wenn ein untergeordneter Baustein seine Aktivitäten beendet hat. Im allgemeinen haben die priorisierteren Bausteine die ständigen Vorrechte. Damit ein untergeordneter Baustein auch einmal zu Wort kommt, gibt des den Eingang ANYRQST, mit dessen Hilfe er seine Ansprüche anmelden kann.

Für eine Prioritätsdekodierung existieren zwei Techniken:

1. Die parallele Prioritätsbestimmung und
2. die serielle Prioritätsbestimmung.

Für beide Möglichkeiten sind die Bausteine vorbereitet.

3.3.3.4.1 Die parallele Prioritätsfestlegung

In der parallelen Prioritätsfestlegung wird eine separate Bus-Anforderungsleitung $\overline{\text{BREQ}}$ für jeden Busverwalter benötigt (Bild 3-46). Jede $\overline{\text{BREQ}}$-Leitung ist an den Eingang eines Prioritätsenkoders geführt, der die binäre Adresse der aktiven $\overline{\text{BREQ}}$-Leitung mit er höchsten Priorität ausgibt. Die Adresse wird anschließend dekodiert, um die entsprechende Bus-Priority-In-$\overline{\text{BPRN}}$-Leitung auszuwählen. Der Busverwalter, der auf diese Art freigegeben wurde, gestattet nun dem Prozessor den Zugriff auf den Systembus. Mit dem Signal $\overline{\text{BUSY}}$ wird anderen priorisierteren Bausteinen angezeigt, daß der Systembus benötigt wird und noch nicht frei ist.

3.3.3.4.2 Die serielle Prioritätsfestlegung

Durch Verkettung der Bausteine benötigt die serielle Prioritätsfestlegung kein Enkoder-Dekoder-Paar. Zu diesem Zweck wird der Bus-Priority-$\overline{\text{BPRO}}$-Ausgang an den Bus-Priority-$\overline{\text{BPRN}}$-Eingang des untergeordneten Bausteins geführt (Bild 3-47).

Für welche Art man sich letztendlich entscheidet, hängt von der Zahl der Busverwalter ab. Sie wird in der seriellen Verkettung durch die Frequenz des $\overline{\text{BCLK}}$-Signals und der Durchlaufverzögerung der Bausteine begrenzt. So können beispielsweise bei 10 MHz nur drei Bausteine kaskadiert werden. Ist das Multi-Prozessor-System größer, muß die parallele Methode verwendet werden.

In den Bildern 3-48 und 3-49 sind Beispiele für den praktischen Einsatz des 8289 gegeben.

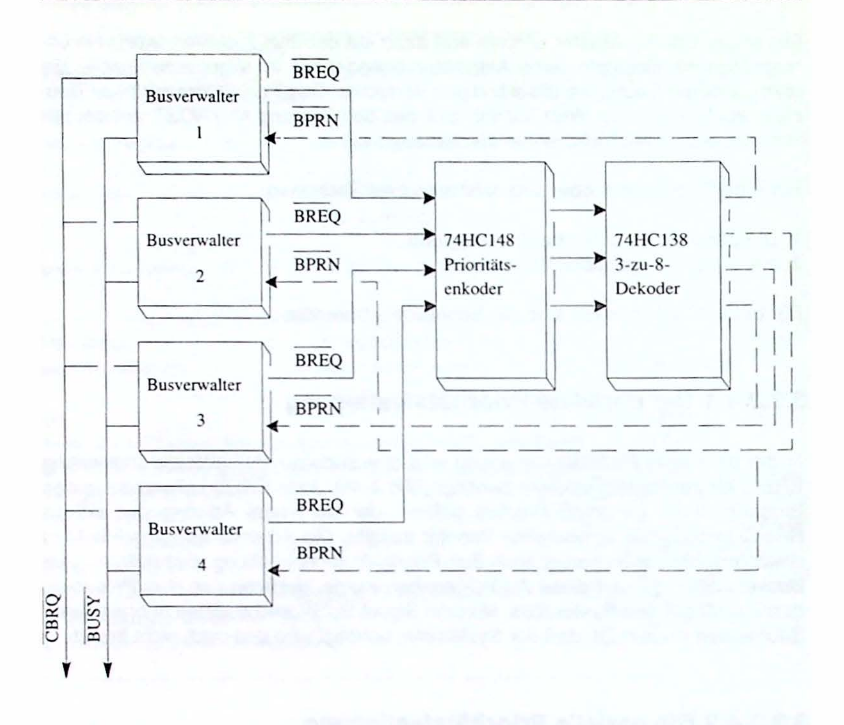


Bild 3-46. Die parallele Prioritätsfestlegung

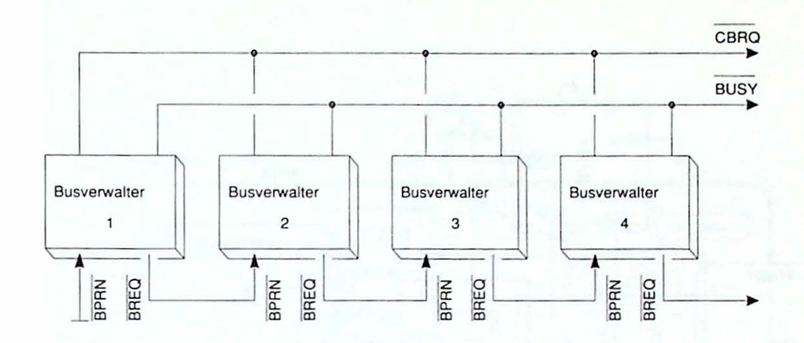


Bild 3-47. Die serielle Prioritätsfestlegung

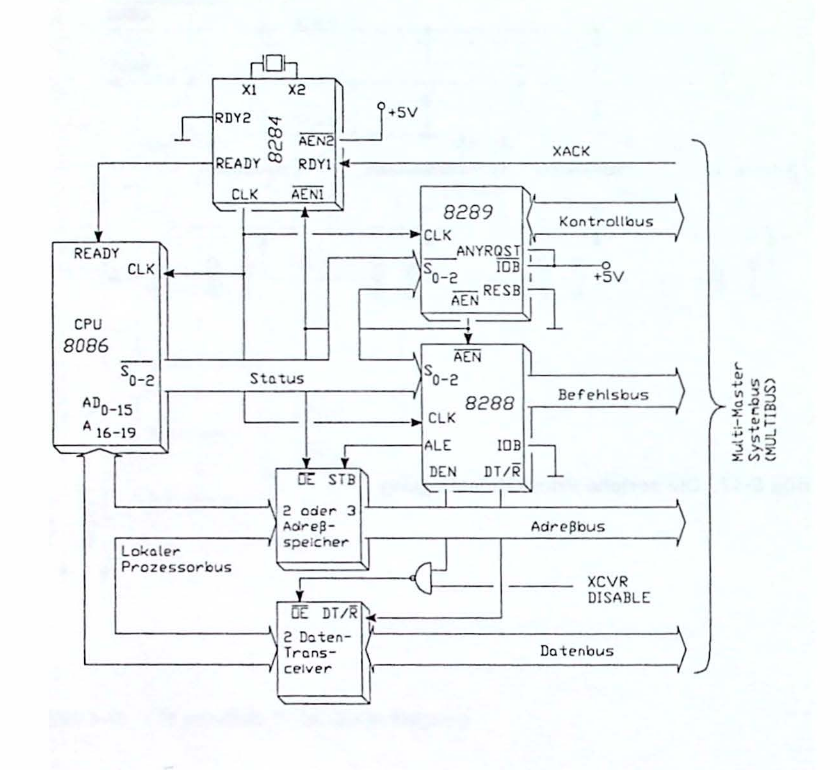


Bild 3-48. Typisches CPU-System mittlerer Komplexität mit dem Busverwalter 8289

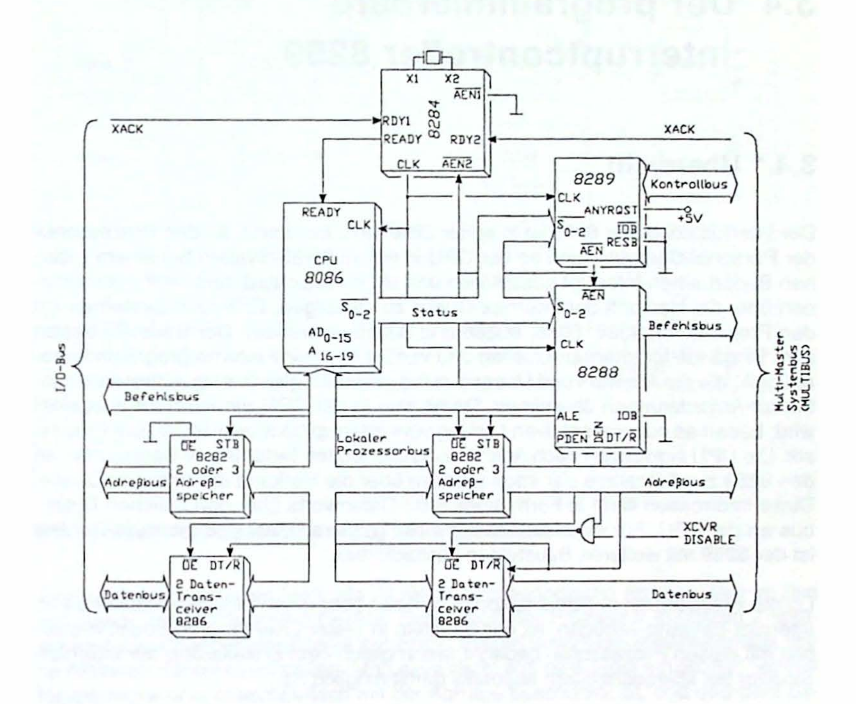


Bild 3-49. Typisches I/O-Bus-System mittlerer Komplexität mit dem Busverwalter 8289

3.4 Der programmierbare Interruptcontroller 8259

3.4.1 Übersicht

Der Interruptcontroller 8259 ist in erster Linie dazu bestimmt, an den Prozessoren der Personal-Computer und an der CPU in einem 80/85-System bei einem externen Bedarf einen Interrupt auszulösen und sie im Anschluß daran mit Informationen über die Herkunft der Interrupt-Quelle zu versorgen. Er wird in Systemen mit den Prozessoren 8088, 8086, 80286 und 80386 verwendet. Der Baustein besitzt acht Eingänge für Interrupt-Quellen und verfügt über eine interne programmierbare Logik, die die Auswahl und Überwachung zwischen gleichzeitig auftretenden Interrupt-Anforderungen übernimmt. Damit also in der CPU ein Interrupt ausgelöst wird, bedarf es nur einer aktiven Leitung vom Interruptcontroller 8259 zum Prozessor. Die CPU signalisiert nach Abarbeitung des letzten Befehls ihre Bereitschaft an den 8259 zur Aufnahme der Informationen über die Herkunft der Interrupt-Quelle. Diese Information fließt in Form eines 8-Bit-Datenworts über den üblichen Datenbus an die CPU. Für die Handhabung einer größeren Zahl von Interrupt-Quellen ist der 8259 mit weiteren Bausteinen kaskadierbar.

Da die Prozessoren in Personalcomputern nur über einen einzigen maskierbaren Interrupt-Eingang verfügen, ist der Baustein in erster Linie für eine Zusammenarbeit mit diesen Prozessoren gedacht, wenn gleich eine Erweiterung der Interrupt-Struktur bei Mikrocontrollern ebenfalls damit möglich ist.

3.4.1.1 Interrupts in Computersystemen

Die Entwicklung eines Computersystems verlangt die angemessene Berücksichtigung der Belange von Peripheriegeräten wie Tastatur, Anzeigen, Sensoren, Diskettenstationen und weiteren Komponenten, so daß das ganze System auf befriedigende Weise seine Aufgabe erfüllen kann.

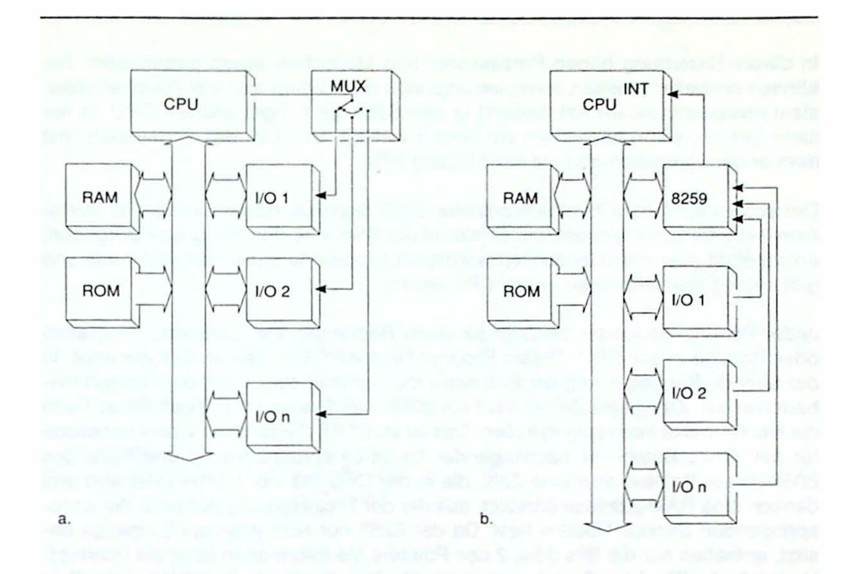


Bild 3-50 a. Abfragemethode; b. Interrupt-Methode

Eine der herkömmlichen Methode ist die zeitlich periodische Abfrage aller an den Prozessor angeschlossener Geräte. Dabei muß der Prozessor in kurzen Zeitabständen alle Peripheriebausteine auf ihren Zustand testen, um gegebenenfalls seine Aktionen darauf einzustellen. Man sieht leicht ein, daß dabei ein großer Teil des Hauptprogramms ausschließlich mit der Abfrage beschäftigt ist, und daß dem System weniger Zeit für die eigentliche Aufgabe zur Verfügung steht.

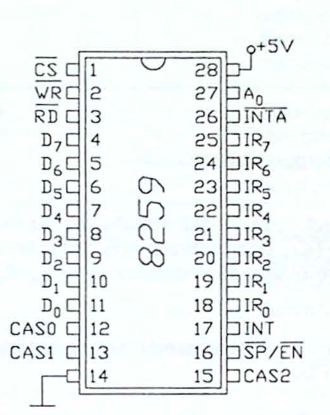
Ein Vergleich aus dem täglichen Leben verdeutlicht das. Um festzustellen, ob ein Gast die Wohnung betreten will, müßte man in regelmäßigen Zeitabständen, z.B. alle zwei Minuten, vor der Wohnungstür schauen, ob jemand da ist. Das ist sehr lästig. Zum Glück hat man eine Klingel und kann so unbesorgt seiner Hauptbeschäftigung nachgehen, bis der Klingelton anzeigt, daß jemand vor der Tür steht. Die Klingel erzeugt also einen Interrupt, der wesentlich exakter mitteilt, wann die Tür geöffnet werden soll. Das lästige Nachsehen entfällt, und die eigene Arbeit kann effektiver ausgeführt werden.

In dieser Beziehung haben Prozessoren und Menschen etwas gemeinsam: Sie können schneller arbeiten, wenn sie ungestört sind. Wenn also ein Peripheriebaustein etwas mitzuteilen hat, benutzt er den 8259 als Klingel und die CPU ist nur dann gestört, wenn tatsächlich ein Bedarf vorliegt. Das Ohr des Prozessors, mit dem er den Klingelton hört, ist der Eingang INT.

Der programmierbare Interruptcontroller 8259 dient als Hauptmanager in Systemen mit Interrupt-Philosophien. Er nimmt die Wünsche der Peripherie entgegen, entscheidet über die Dringlichkeit, vergleicht mit gerade laufenden Interrupts und gibt seine Entscheidungen an die CPU weiter.

Jeder Peripheriebaustein benötigt für seine Bedienung ein spezielles Programm oder Routine in der CPU. Dieser Programmteil wird Service-Routine genannt. In der Service-Routine nimmt der Prozessor die Kommunikation mit dem Peripheriebaustein auf. Zu diesem Zweck muß der 8259 dem Prozessor in irgendeiner Form die Herkunft des Interrupts mitteilen. Das ist im 80/85-System der Operationskode für den CALL-Befehl mit nachfolgender 16-Bit-Einsprungadresse und Falle des 8086/88 ein Pointer, also eine Zahl, die in der CPU mit vier multipliziert wird und danach eine RAM-Adresse darstellt, aus der der Prozessor die Adresse der anzuspringenden Service-Routine liest. Da der 8259 nur acht Interrupt-Eingänge besitzt, enthalten nur die Bits 0 bis 2 des Pointers die Information über die Interrupt-Herkunft, die Bits 3 bis 7 sind voreingestellte Bits, die durch die CPU bei der Programmierung des Bausteins gesetzt werden. Somit können die Interrupt-Vektoren im RAM nur an den Adressen von 0 bis $3FF_h$ liegen:

Interrupt-Pointer:	x x x x x	i i i
	Programmierte Bits	Nummer des Interrupts



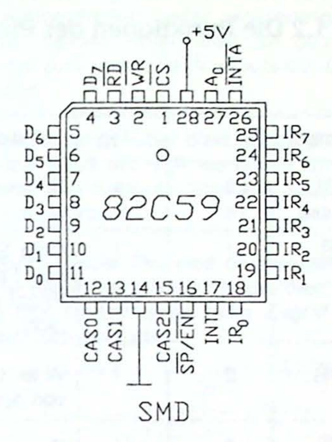


Bild 3-51. Pin-Belegung des 8259

3.4.1.2 Die Funktionen der Pins

Symbol	Pin Nr.	Typ	Funktion
+5 V, Masse	28 14	I I	Stromversorgungsanschlüsse
$\overline{CS}$	1	I	Chip Select. Low-Pegel erlaubt die Aufnahme von Lese-$\overline{RD}$- und Schreib-$\overline{WR}$-Befehle. High-Pegel unterbindet die Interrupt-Kommunikation nicht.
$\overline{WR}$	2	I	Write. Low-Pegel signalisiert einen Schreibbefehl von der CPU.
$\overline{RD}$	3	I	Read. Low-Pegel signalisiert einen Lesebefehl von der CPU.
D_7-D_0	4-11	I/O	Bi-direktionaler Datenbus. Daten und Befehle werden mit diesen Leitungen übermittelt.
CAS_0- CAS_2	12, 13, 15	I/O	Kaskadierung. In einem Master sind sie Ausgänge,in einem Slave Eingänge. Über sie adressiert der Master einen Slave (maximal 8). Ist keine Kaskadierung erwünscht, bleiben die Leitungen offen.
$\overline{SP}/\overline{EN}$	16	I/O	Slave Program Input/Enable Buffer Output. Üblicherweise wird für ein Master an diesem Eingang High-Pegel und für einen Slave Low-Pegel angelegt. Wird die Master/Slave-Struktur durch die Software festgelegt, ist dieser Pin Ausgang und dient zur Freigabe eines Bustreibers.
INT	17	O	Interrupt. Dieser Ausgang wird zum gleichnamigen Eingang am Prozessor geführt und ist high-aktiv, wenn der 8259 einen Interrupt auslösen will.

IR7 - IR_0	18 - 25	I	Interrupt Request. High-Pegel löst an diesen Eingängen einen Interrupt aus. Das kann je nach Programmierung flanken- oder pegelgetriggert erfolgen. In beiden Fällen muß die positive Spannung bis zum ersten $\overline{\text{INTA}}$-Impuls der CPU stabil anliegen.
$\overline{\text{INTA}}$	26	I	Interrupt Acknowledge. Über diese Leitung zeigt die CPU dem 8259 an, daß sie bereit ist und nachfolgend Informationen über die Herkunft des Interrupts wünscht.
A0	27	I	Adreßeingang. Dieser Pin wird normalerweise mit einer Adreßleitung verbunden und dient zur Steuerung des Datenflusses beim Zugriff auf Befehls- oder Statusregister.

3.4.1.3 Der Interrupt-Ablauf

Der Interruptcontroller 8259 ist in vielfältiger Weise auf die Belange spezieller Systeme programmierbar, so daß ein flüchtiger Überblick über seine zahlreichen Möglichkeiten zunächst mehr Verwirrung als Klarheit schafft. Hauptaufgabe des Bausteins ist es, die eingehenden Interrupt-Anforderungen zu sammeln, sie nach ihrer Wichtigkeit zu sortieren und der CPU Mitteilung über die Herkunft zu machen. Dieses Vorgehen wird Interrupt-Sequenz genannt.

Die Interrupt-Sequenz erfolgt nach zwei Arten, wobei der Programmierer sich für eine von beiden bei der Initialisierung des Bausteins entscheiden muß. Es ist dies der 8088- bzw. der 8085-Modus. Der 8088-Modus wird in Systemen mit den Prozessoren 8086, 8088, 80286 und 80386 und deren CMOS-Versionen verwendet, der 8085-Modus in Zusammenarbeit mit den Prozessoren 8080 und 8085.

Die Interrupt-Kommunikation mit der CPU erfolgt in mehreren Schritten:

3.4.1.3.1. Der 8088-Modus

(1) Gehen einer oder mehrere Interrupt-Eingänge an High-Pegel, werden die entsprechenden Flip-Flops im Interrupt-Anforderungsregister gesetzt.

(2) Nach Prüfung auf Maskierungen und Prioritäten sendet der 8259 über die Leitung INT ein Signal an die CPU.

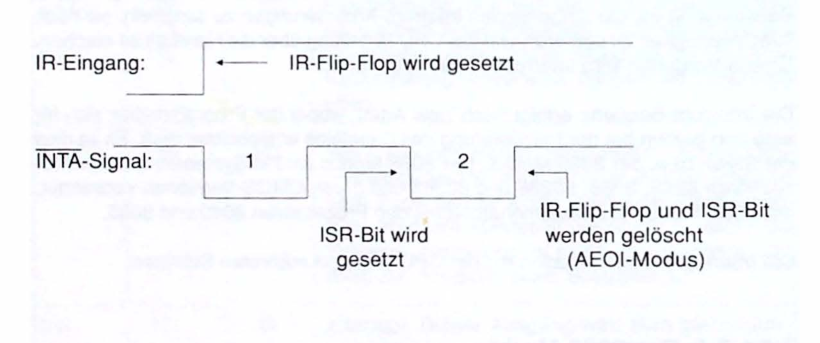
(3) Als Antwort und Anerkennung der Anforderung gibt die CPU ihrerseits über die Leitung $\overline{INTA}$ ein Signal an den 8259.

(4) Mit dem Eingang des ersten $\overline{INTA}$-Impulses werden noch keine Daten an die CPU übermittelt. Der Bus bleibt im Tri-State. Der Interruptcontroller benutzt diesen Impuls vielmehr zum Takten interner Vorgänge.

(5) Die CPU gibt in ihrem folgenden Maschinenzyklus einen zweiten $\overline{INTA}$-Impuls aus. Dabei wird vom 8259 ein 8-Bit-Pointer auf den Bus gelegt, der von der CPU gelesen und dekodiert wird. Der zweite $\overline{INTA}$-Impuls hat also die gleiche Wirkung wie ein Lesezugriff auf einen RAM-Baustein.

(6) Nach diesem Vorgang ist der Interrupt-Zyklus beendet und die CPU springt in die entsprechende Interrupt-Routine. Ist der 8259 im Automatic-End-Of-Interrupt-Modus AEOI, wird das betreffende Bit im In Service Register ISR gelöscht. In den anderen Modi bleibt das Bit gesetzt, bis von der CPU ein End-Of-Interrupt-Befehl EOI eintrifft.

Signalformen bei der Ausführung des Interrupts:



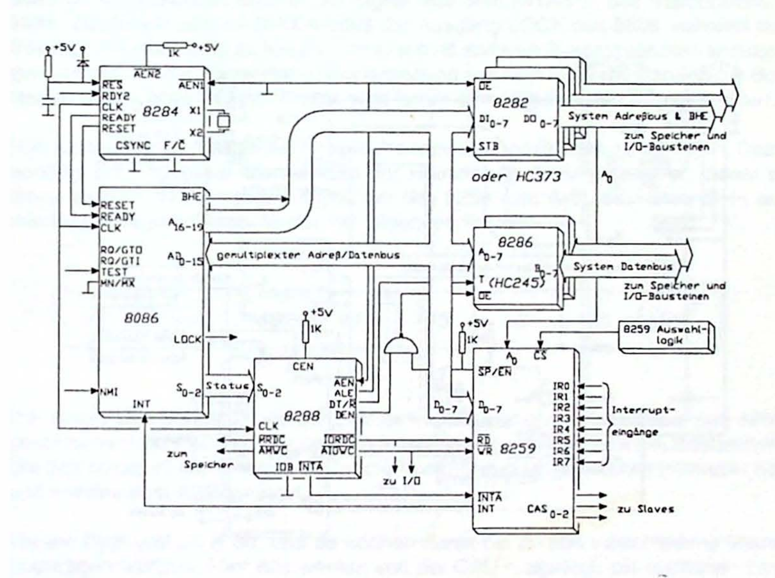


Bild 3-52. Der 8259 in einem 8086-System

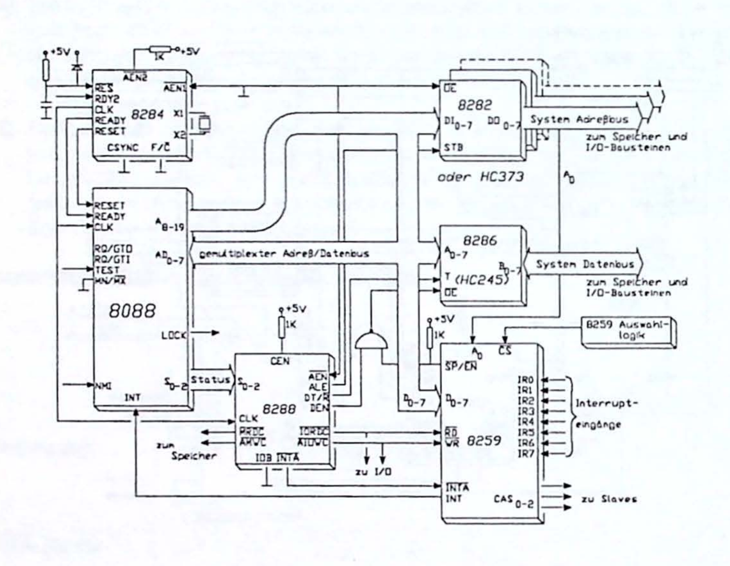


Bild 3-53. Der 8259 in einem 8088-System

Wenn ein Interrupt bei der CPU durch einen INT-Impuls angemeldet wird und Interrupts freigegeben sind, führt der 8088 einen speziellen Maschinenzyklus aus, die Interrupt-Anerkennung. In diesem Maschinenzyklus gibt der Prozessor das komplette Flag-Register auf den Stack und löscht das Interrupt-Flip-Flop, um weitere Unterbrechungen auszuschließen. Anschließend werden das Code-Segment und der Instruction-Pointer ebenfalls auf dem Stack abgelegt. Somit sind die Informationen über den ursprünglichen Flagzustand und die Return-Adresse auf dem Stack gerettet. Jetzt gibt der Prozessor den ersten $\overline{INTA}$-Impuls an den 8259 aus. Ist der 8088 im MIN-Modus kommt das Signal aus dem gleichnamigen Pin, ist der 8088 im MAX-Modus, kommt das Signal aus dem $\overline{INTA}$-Pin des Buscontrollers 8288. Zusätzlich geht im MAX-Modus der Ausgang LOCK des 8088 während der Interrupt-Anerkennung an Masse, um eventuell anderen Busprozessoren anzuzeigen, daß sie für die Dauer der Vektorauslesung aus dem 8259 die Hände vom Bus lassen sollen. Mit LOCK an Masse wird ferner eine HOLD-Anforderung ignoriert.

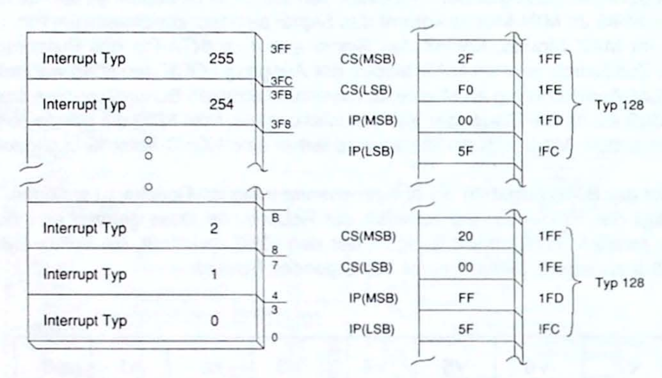
Nun ist der 8088 bereit, in die entsprechende Interrupt-Routine zu springen. Dazu benötigt der Prozessor die Adresse der Routine. An diese gelangt er, indem er einen zweiten $\overline{INTA}$-Impuls ausgibt, der den 8259 veranlaßt, ein Vektor-Byte auf den Bus zu legen. Dieser Vektor hat folgendes Format:

V7	V6	V5	V4	V3	n2	n1	n0

Die Vektor-Bits V3 bis V7 werden bei der Initialisierung von der CPU in den 8259 geschrieben und sind bis zur nächsten Initialisierung unveränderlicher Bestandteil. Die Bits n0 bis n2 stellen binärkodiert die Nummer des aufgetretenen Interrupts dar und werden vom 8259 gesetzt.

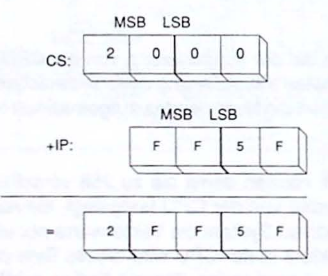
Dieses Byte umfaßt 8 Bit, und es können damit bis zu 256 verschiedene Werte übertragen werden. Fünf Bits werden von der CPU festgelegt, die restlichen vom 8259 gebildet, so daß in einem aktiven System die Vektorwerte nur um 4 schwanken. Nach dem Eintreffen des Vektors in der CPU wird dieses Byte zweimal nach links geschoben und die neu dazukommenden Bits mit Nullen gefüllt, was einer Multiplikation mit vier entspricht. Nun zeigen die Werte in Viererschritten auf die absoluten Adressen von 000 bis $3FC_h$ oder 0 bis 1020_d, die die Interrupt-Vektortabelle des 8088 bilden (Bild 354 a). Zu beachten ist ferner, daß die Adressen 0 bis $7F_h$, um Kompatibilität zu gewährleisten, nicht von dem 8259 benutzt werden sollen, da dieser Bereich für bestehende und künftige Hard- und Software-Produkte der Firma Intel reserviert ist. Aus dieser Vektortabelle entnimmt nun der Prozessor die Startadresse der Interrupt-Routine, und zwar steht in den beiden ersten Bytes der Inhalt des Instruction-Pointers mit Low-Byte zuerst gefolgt von High-Byte. Im

dritten und vierten Speicherplatz ist der Inhalt des Code-Segments zu finden, ebenfalls in der Reihenfolge Low-Byte, High-Byte. In Bild 354 b sind zwei Beispiele für ein und dieselbe Sprungadresse gegeben. Der Prozessor ermittelt daraus die 20-Bit-Adresse, indem er zum Instruction-Pointer das um vier Bits nach links geschobene Code-Segment addiert (Bild 3-54 c).



a. 8088 Interrupt-Vektor-Tabelle

b. Identische Sprungadressen



c. Berechnung der Sprungadresse

Bild 3-54 Die Handhabung des Vektor-Byte durch den Prozessor

Nachdem die Interrupt-Routine beendet ist, springt die CPU wieder in das Hauptprogramm zurück, indem sie einen IRET-Befehl (Interrupt Return) ausgibt. Der IRET-Befehl holt die alte Adresse von Stack und stellt die ursprünglichen Flag-Inhalte wieder her. Insbesondere schließt das auch den Zustand des Interrupt-Flags IF ein. Somit sind Interrupts automatisch nach einem IRET-Befehl freigegeben. Will das Programm während der Interrupt-Routine weitere Interrupts aufnehmen, ist mit dem Befehl Enable-Interrupt EI das Interrupt Flag IF zu setzen.

Neben der Erzeugung externer Interrupts über den Pin INT, kann der 8088 interne Interrupts mittels Software auslösen. Drei solcher Befehle kennt der 8088:

1. INT: INT ist ein Zwei-Byte-Befehl, der die gleiche Wirkung hat, als hätte der 8259 einen externen Interrupt ausgelöst. Im zweiten Byte steht nämlich die gleiche Vektoradresse, wie sie über den 8259 ausgegeben wird.

2. INT Typ 3: Es ist ein Ein-Byte-Befehl, der den Interrupt von Typ 3 wählt. (Die Adresse ist im RAM an der Stelle 0B zu finden.)

3. INTO: INTO ist ein bedingter Ein-Byte-Befehl von Typ 4, dessen Ausführung davon abhängt, ob das Overflow-Flag OF gesetzt ist.

Von der Ausführung der Interrupt-Routine unterscheiden sich software-initiierte durch nichts von hardware-ausgelösten Interrupts.

3.4.1.3.2. Der 8085-Modus

(1) Gehen einer oder mehrere Interrupt-Eingänge an High-Pegel, werden die entsprechenden Flip-Flops im Interrupt-Anforderungsregister IRR gesetzt.

(2) Nach Prüfung auf Maskierungen und Prioritäten sendet der 8259 über die Leitung INT ein Signal an die CPU.

(3) Als Antwort und Anerkennung der Anforderung gibt die CPU ihrerseits über die Leitung $\overline{INTA}$ ein Signal an den 8259.

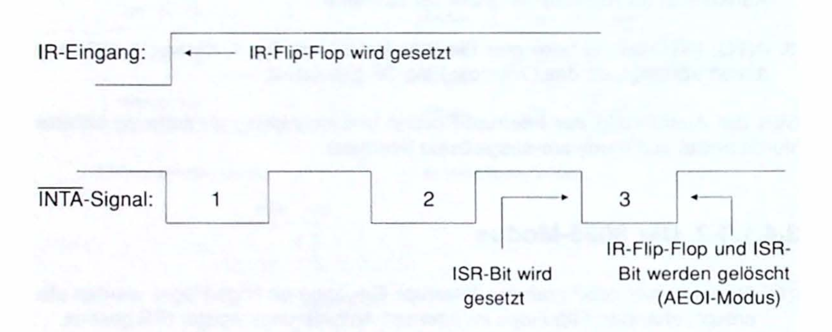
(4) Mit dem Eintreffen des ersten $\overline{INTA}$-Impulses von der CPU, wird der Operationskode des CALL-Befehls (CD_h) für den 8085 auf den Bus gelegt. Die CPU liest somit einen Maschinenbefehl aus dem 8259!

(5) Da der CALL-Befehl ein Drei-Byte-Befehl ist, müssen somit noch weitere zwei $\overline{INTA}$-Signale eingehen, damit die CPU die entsprechende Interrupt-Einsprungadresse erfährt.

(6) Die beiden folgenden $\overline{INTA}$-Signale veranlassen den 8259, die Sprungadresse auszugeben, und zwar zuerst das Low-Byte gefolgt vom High-Byte. Anders als im 8088-Modus handelt es sich hierbei um eine absolute Adresse und nicht um einen Vektor. Das dem IR-Eingang entsprechende In-Service-Bit ISR wird mit Beginn des einkommenden dritten $\overline{INTA}$-Impulses gesetzt.

(7) Damit ist die Ausgabe des Drei-Byte-CALL-Befehls beendet, und die CPU führt den Interrupt aus. Mit der steigenden Flanke des dritten $\overline{INTA}$-Impulses wird das ISR-Bit gelöscht, sofern der 8259 sich im Automatic-End-Of-Interrupt-Modus befindet. Ansonsten muß die CPU den speziellen End-Of-Interrupt-Befehl senden, da sonst bei gesetztem ISR-Bit keine neueingehende Interrupt-Anforderung akzeptiert würde.

Signalformen bei der Ausführung des Interrupts:



Die Hardware-Verbindung des 8259 mit dem Prozessor 8080 ist in Bild 3-55 gezeigt und die Einbindung in ein 8085-System in Bild 3-56.

Fordert ein Peripheriebaustein einen Interrupt an, legt er einen Eingang IR am 8259 an Plus. Wenn der 8259 diese Anforderung akzeptiert, was von seiner Programmierung abhängt, setzt er die Leitung INT auf High-Pegel. Am 8080/85 kann jederzeit ein Interrupt angemeldet werden, es muß dabei auf keine Synchronisierung mit dem Takt geachtet werden. Ob der Prozessor allerdings diesen Interrupt aufnimmt, hängt von der Software ab. Der Interrupt kann per Befehl gesperrt (DI, Disable Interrupt) oder freigegeben (EI, Enable Interrupt) werden. Durch beide

Befehle wird der Zustand eines Flip-Flops verändert, das die interne INT-Leitung unterbindet oder ermöglicht. Der 8080 zeigt diesen Zustand über den Ausgang INTE an.

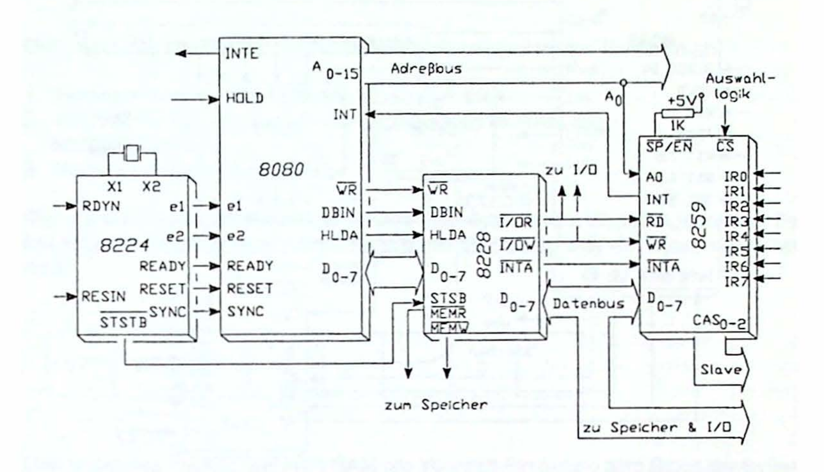


Bild 3-55. Standardkonfiguration mit dem 8080

Am Ende eines jeden Maschinenzyklus wird der Zustand des INT-Pins bestimmt. Wenn ein Interrupt anliegt und freigegeben ist, führt der 8080/85 einen Interrupt-Maschinenzyklus aus, währenddessen das Interrupt-Freigabe-Flip-Flop gelöscht wird, um weitere Interrupts zu verhindern, bis ein EI-Befehl ausgeführt wird. Anders als in normalen Maschinenzyklen wird hierbei der Programmzähler nicht erhöht. Dann gibt der 8080 über den Systemkontroller 8228, der 8085 direkt an den 8259 einen $\overline{INTA}$-Impuls aus, um ihm mitzuteilen, daß er bereit ist, die Interrup-Routine auszuführen.

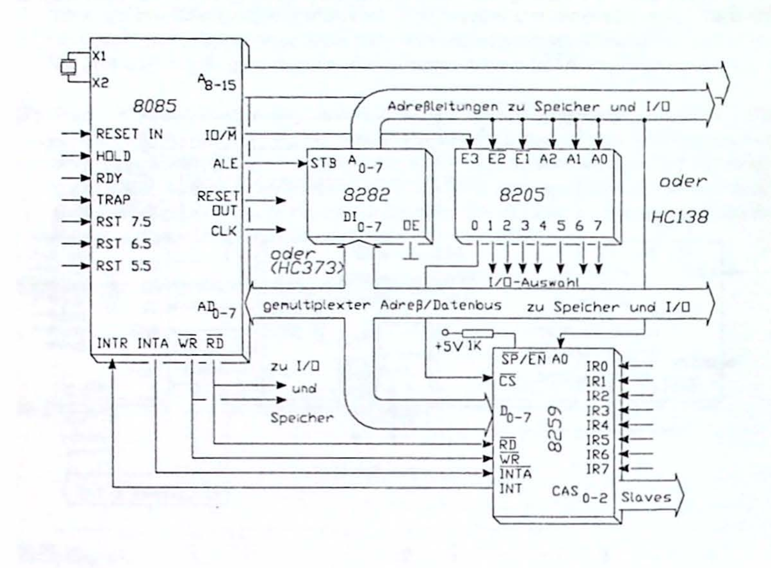
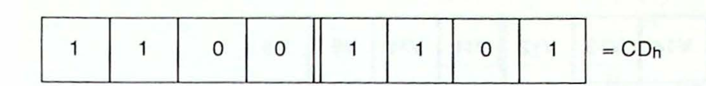


Bild 3-56. Standardkonfiguration mit dem 8085

Der 8259 kann dem Prozessor nun die Einsprungadresse für den Interrupt mitteilen. Das geschieht mit einer Sequenz von drei $\overline{INTA}$-Impulsen, die der Prozessor über den 8228 ausgibt. Mit dem ersten $\overline{INTA}$-Impuls gibt der 8259 den Operationskode für einen CALL-Befehl auf den Bus. Das bewirkt, daß der aktuelle Inhalt des Programmzählers auf dem Stack abgelegt wird. Mit dem zweiten $\overline{INTA}$-Impuls gibt der 8259 das Adressen-Low-Byte (LSB) und mit dem dritten $\overline{INTA}$-Impuls das Adressen-High-Byte (MSB) auf den Bus. Nach Beendigung dieser Sequenz springt die CPU an die so erhaltene Adresse, um den Interrupt zu bearbeiten.

Das erste Interrupt-Vektor-Byte hat immer folgendes Format und stellt in der Mnemonik des 8080/85 einen CALL-Befehl dar:

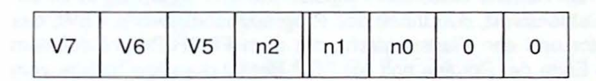


1	1	0	0	1	1	0	1	= CD_h

Der Inhalt des zweiten Interrupt-Vektor-Bytes hängt von drei Faktoren ab:

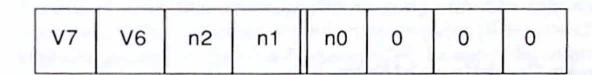
1. Initialisierungswert der Basisadresse
2. Intervall für die Interrupt-Einsprungadressen im RAM des 8080/85-Systems.
3. Nummer der Interrupt-Quelle

Das zweite Interrupt-Vektor-Byte ist das Low-Byte für die Einsprungadresse. Es hat folgendes Format, wenn im RAM pro Interrupt-Einsprung vier Bytes reserviert sind:



V7	V6	V5	n2	n1	n0	0	0

Und folgendes Format, wenn im RAM pro Interrupt-Einsprung acht Bytes reserviert sind:



V7	V6	n2	n1	n0	0	0	0

Die Bits V5 bis V7 werden bei der Initialisierung durch die CPU in den 8259 geschrieben und bei der Adreßausgabe übernommen; die Bits n0 bis n2 stellen in binärer Form die Nummer des auslösenden Interrupts dar und die beiden bzw. drei restlichen Bits sind immer Nullen. Somit müssen die Interrupt-Einsprungadressen im RAM entweder immer in Vierer- oder in Achterschritten erfolgen. Die Wahl des Intervalls erfolgt im Initialisierungswort 1 mit Bit 2.

Das dritte Interrupt-Vektor-Byte bildet das Adressen-High-Byte und wird ebenfalls im 8259 bei seiner Initialisierung durch die CPU gesetzt. Es hat folgende Form:

V15	V14	V13	V12	V11	V10	V9	V8

Haben die Bits V15 bis V5 den Inhalt: 0101 1011 011 und wird ein Interrupt der Nummer drei gemeldet, gibt der 8259 bei dem Intervall 4 folgende Sprungadresse an den Prozessor: 0101 1011 011**0** **11**00 = 5B 6C

Mit dem Intervall 8 lautet die Sprungadresse: 0101 1011 01**01** **1**000 = 5B 58.

Da nach vier bzw. acht Bytes im RAM bereits die nächste Interrupt-Einsprungadresse folgt, werden im allgemeinen Sprungbefehle zur eigentlichen Routine an diesen Stellen stehen. Ein 4-Byte-Intervall wird man dann wählen, wenn man eine möglichst kompakte Sprungtabelle haben möchte.

Werden in der Interrupt-Routine dieselben Register wie im Hauptprogramm benutzt, so ist es empfehlenswert, den Inhalt des Programmstatusworts PSW, das aus dem Akkumulator und den Flags besteht, mit dem PUSH-Befehl auf dem Stack zu retten. Am Ende der Routine holt ein POP-Befehl die alten Inhalte vom Stack und läßt das Hauptprogramm mit den ehemaligen Zuständen fortfahren. Abgeschlossen wird der Interrupt mit einem normalen RET-Befehl, der den Programmzähler mit dem alten Inhalt lädt. Bemerkt sei hier, daß der RET-Befehl nicht automatisch neue Interrupts freigibt. Das muß ausdrücklich durch einen EI-Befehl entweder am Ende der Interrupt-Routine oder an geeigneter Stelle im Hauptprogramm stattfinden.

3.4.1.4 Die internen Funktionsblöcke

Das Blockdiagramm des 8259 ist in Bild 3-57 gezeigt. Wie die Abbildung zeigt, besteht der 8259 aus acht Blöcken: dem Interrupt-Anforderungsregister (Interrupt Request Register, IRR), dem In Service-Register (ISR), dem Interrupt-Maskenregister (IMR), dem Prioritätsdekoder, der Kaskadierungseinheit, dem Puffer für Datenein/ausgabe und der Schreib/Leselogik. Nur die ersten fünf Blocks haben direkt etwas mit der Interrupt-Verarbeitung zu tun.

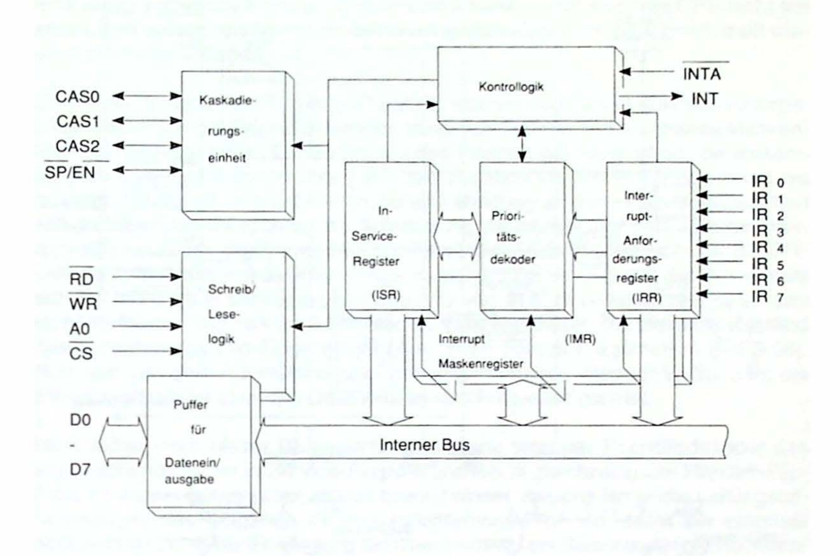


Bild 3-57. Blockdiagramm des 8259

Prinzipiell wird eine Interrupt-Anforderung von den drei kaskadierten Registern verarbeitet: Das Interrupt-Anforderungsregister (IRR) speichert alle eingehenden High-Pegel, das In-Service-Register (ISR) bewahrt die Interrupt-Level auf, die gerade von der CPU in Bearbeitung sind, und Interrupt-Maskenregister (IMR) speichert die Bits der Interrupt-Linien, die gesperrt werden sollen. Zwischen dem IRR und ISR liegt der Prioritätsdekoder, dessen Hauptaufgabe es ist, über die Dringlichkeit von mehreren eintreffenden Interrupt-Anforderungen zu entscheiden. Bild 3-58 zeigt schematisch die Handhabung einer Interrupt-Anforderung und die Zusammenwirkung der verschiedenen Register. Im Bild ist eine der insgesamt acht Prioritätszellen dargestellt.

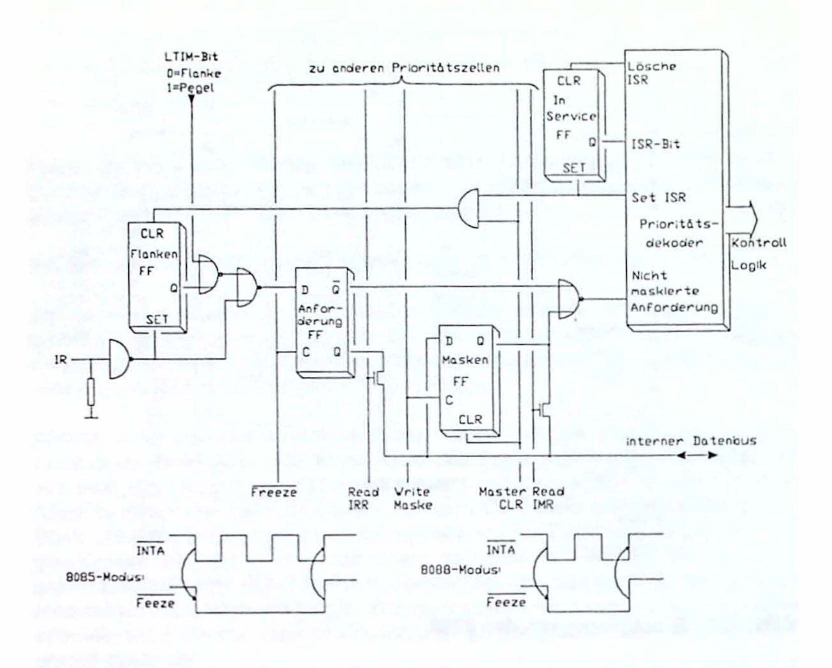


Bild 3-58. Zelle des Prioritätsdekoders

Der beste Weg, die Arbeitsweise einer Prioritätszelle zu verstehen, ist das Nachvollziehen der internen Verarbeitungssequenz, wenn ein Interrupt am Eingang erscheint. Prinzipiell kann das Signal flankengetriggert oder pegelsensitiv weiterverarbeitet werden. Welche Methode verwendet wird, hängt von der speziellen Anwendung ab. Liegt keine Interrupt-Anforderung an, ist der Eingang IR über den internen Pull-Down an Masse und das Flanken-Flip-Flop ist über den Inverter gesetzt. Wenn die Flankentriggerung gewählt ist, gibt der Ausgang Q des Flanken-Flip-Flops über das zweite NOR-Gatter den Zugang zum Anforderungs-Flip-Flop frei. Dieses Eingangs-Gate wird gesperrt, sobald der Eingang IR an High-Pegel liegt und die Interrupt-Anforderung durch ein $\overline{\text{INTA}}$-Signal quittiert wurde. Das verhindert solange eine weitere Interrupt-Auslösung, bis der Pegel am Eingang IR wieder an Masse liegt und das Flanken-Flip-Flop erneut setzt. Ist der pegelsensitive Modus gewählt, ist der Q-Ausgang des Flanken-Flip-Flops ohne Bedeutung,

d.h. der Pegel am Eingang IR ist vollständig für die Interrupt-Kontrolle zuständig; eine weitere Interrupt-Erzeugung wird nach Anerkennung durch die CPU nicht gesperrt, und spätestens beim Erscheinen eines End-Of-Interrupts-Befehls muß wieder Low-Pegel anliegen.

Sobald ein Interrupt am Eingang IR auftritt, wandert das Signal über das Anforderungs-Flip-Flop zum Prioritätsdekoder, vorausgesetzt das entsprechende Masken-Flip-Flop ist nicht gesetzt. Die Aufgabe des Prioritätsdekoders ist es, die einkommenden Interrupt-Anforderungen mit den augenblicklichen In-Service-Interrupts zu vergleichen und zu entscheiden, ob eine Meldung an den Prozessor ausgehen soll. In folgenden Beispiel sei die Anforderung die einzige und kein weiterer Interrupt sei eingeleitet, dann veranlaßt der Prioritätsdekoder die Kontrollogik, die INT-Leitung zum Prozessor auf High-Pegel zu ziehen. Bei der Antwort der Prozessors auf den INT-Impuls sendet er eine Sequenz von $\overline{\text{INTA}}$-Impulse an den 8259, drei im 8085-Modus und zwei im 8088-Modus. Während dieser Zeit bleibt der Zustand des Anforderungs-Flip-Flops eingefroren (siehe Freeze-Diagramm in Bild 3-58). Nun wird von dem Prioritätsdekoder der entsprechende Interrupt-Vektor bzw. die Einsprungadresse über den Datenbus an den Prozessor geleitet.

Unmittelbar nach dieser Beantwortungssequenz setzt der Prioritätsdekoder das entsprechende Bit im In-Service-Register und löscht gleichzeitig das Flanken-Flip-Flop. Im flankengetriggerten Modus bewirkt dieser Vorgang ferner die Leitungsunterbrechung des Eingangs IR zum Anforderungs-Flip-Flop, damit ein eventuell noch weiterhin aktiver IR-Eingang nicht noch einmal ein Signal durch die Prioritätszelle gibt. Bevor über denselben Eingang erneut ein Interrupt angefordert werden kann, muß er zunächst wieder an Masse gelegt werden. Bei einer Pegeltriggerung bewirkt das Löschen des Flanken-Flip-Flops keine Änderung des Anforderungs-Flip-Flops. Der Zustand des Anforderungs-Flip-Flops wird in diesem Falle ausschließlich vom Zustand des Eingang IR bestimmt. Bleibt der Pegel am IR-Eingang nach dem Löschen des In-Service-Bits weiterhin high, wird sofort ein neuer Interrupt erzeugt. Das In-Service-Register-Bit wird nur durch einen End-Of-Interrupt-Befehl rückgesetzt, der entweder automatisch erfolgt oder von der CPU ausgegeben wird, je nach Initialisierung des 8259.

Ein weiterer Funktionsblock wird vom Datenbus-Puffer gebildet. In der Regel sind seine Pins im Tri-State und erwachen erst zum Leben, wenn Daten vom oder zum Baustein fließen. Damit Daten fließen können, muß der Baustein selektiert, d.h. $\overline{\text{CS}}$ an Masse, und entweder $\overline{\text{RD}}$ oder $\overline{\text{WR}}$ aktiv sein. Es ist dann möglich, Initialisierungs- oder Operationsbefehle in Abhängigkeit von A0 in den Baustein zu schreiben. Ferner kann der Inhalt von jedem der 8-Bit-Register, IRR, ISR und IMR gelesen werden, damit die CPU ihre Aktivitäten nach den Zustand der Register richten kann. Einzelheiten sind an späterer Stelle aufgeführt.

Der dritte Funktionsblock ist die Kaskadierungseinheit. Sie dient in erster Linie dazu, weitere 8259 als Slaves anzubinden zu dem Zweck, die Anzahl von acht Interrupt-Eingängen auf maximal 64 zu erhöhen. In diesem Fall wird die INT-Leitung des Slaves nicht mit der CPU, sondern mit dem Eingang IR des Master 8259 verbunden, wo der Slave wie ein üblicher Peripheriebaustein einen globalen Interrupt auslöst. Auf Grund der Initialisierung weiß der Master, daß auf dieser Leitung ein Slave liegt und sendet über die drei $\overline{\text{CAS}}$-Leitungen seine Adresse aus, um ihn freizugeben. Die nachfolgende Kommunikation erfolgt dann zwischen CPU und Slave. Der Anschluß $\overline{\text{SP}}/\overline{\text{EN}}$ kann benutzt werden, um entweder die Master/Slave-Konfiguration hardware-mäßig zu definieren ($\overline{\text{SP}}/\overline{\text{EN}}$ ist dann Eingang) oder um einen externen Datenpuffer zu steuern, wenn die Vektoradresse auf den Datenbus gelegt wird. Die genaue Verwendung ist bei der Expansion auf 64 Interrupts beschrieben.

3.4.2 Die Betriebsarten des Interruptcontrollers 8259

3.4.2.1 Interrupt Prioritäten

Zahlreiche Modi und Befehle stehen dem Anwender zur Verfügung, um die Interrupt-Selektierung vorzunehmen. Jeder Modus kann von der Software programmiert und jederzeit wieder geändert werden.

Der vollverschachtelte Modus:

Dieser Modus ist die Grundeinstellung des Bausteins. Er muß nicht ausdrücklich programmiert werden, sondern wird bei der Initialisierung automatisch gewählt. Im vollverschachtelten Modus werden eingehende Interrupts ihrer Wichtigkeit nach geordnet, wobei der Eingang IR0 die höchste und der Eingang IR7 die niedrigste Priorität genießt. Eine Auswahl durch die Prioritätslogik findet nur in zwei Fällen statt, wenn nämlich gleichzeitig mehrere Interrupt-Eingänge aktiv werden und wenn bereits Interrupts in Bearbeitung sind. Im ersten Falle wird der Interrupt mit der höchsten Priorität weitergeleitet, im zweiten Falle wird geprüft, ob der einkommende Interrupt höhere Priorität besitzt, als der gerade bearbeitete. Diesem Zweck dient das In-Service-Register ISR, in dem die Informationen über laufende Interrupts abgelegt sind. Handelt es sich um einen Interrupt höherer Priorität, wird der laufende unterbrochen und ein neues INT-Signal an den Prozessor geschickt. Ist die Anforderung gleich oder niedriger, bleibt das entsprechende Bit im Anforderungsregister IRR gesetzt und wird erst nach Beendigung des laufenden Interrupts bedient.

Diese Arbeitsweise kann in gewissen Fällen von Nachteil sein. Meldet ein Peripheriebaustein sehr häufig einen Interrupt an, kann es vorkommen, daß ein Baustein niedrigerer Priorität nicht mehr zu Wort kommt und von der CPU vergessen wird. Deswegen besteht die Möglichkeit, anderen Interrupts die höchste Priorität zuzuweisen. Das geschieht durch Beschreiben des Operationsworts 2 mit dem Bitmuster 1100 0nnn, wobei nnn die binäre Nummer des Interrupt-Eingangs ist, der die höchste Priorität haben soll. Die Prioritäten der nachfolgenden Eingänge verschieben sich in entsprechender Weise.

Beispiel:

Eingang:	IR7	IR6	IR5	IR4	IR3	IR2	IR1	IR0
Priorität	8.	7.	6.	5.	4.	3.	2.	1.
Beschreiben der Operationsworts 2 mit 1100 0101 = C5								
Priorität	3.	2.	1.	8.	7.	6.	5.	4.
Beschreiben der Operationsworts 2 mit 1100 0010 = C2								
Priorität	6.	5.	4.	3.	2.	1.	8.	7.

Tritt nun ein Interrupt höherer Priorität auf, wird erneut eine Meldung an den Prozessor gesendet, der diese jedoch nur dann empfangen kann, wenn er zuvor durch einen Befehl die Unterbrechung durch Interrupts freigegeben hat. Das ist besonders dann zu beachten, wenn der Prozessor sich gerade in einer Interrupt-Routine befindet, weil beim Einsprung in diese Routine automatisch weitere Interrupts gesperrt werden. Es ist daher empfehlenswert, möglichst rasch neue Interrupts wieder zu erlauben, damit ein ranghöheres Ereignis registriert werden kann. Die entsprechenden Befehle lauten für den 8088 STI (Set Interrupt) und für den 8080/85 EI (Enable Interrupt).

Bild 3-59 illustriert den korrekten Gebrauch der entsprechenden Befehle und zeigt die Interrupt-Verschachtelung. In der IR3-Routine tritt zu Beginn eine Interrupt-Anforderung höherer Priorität auf, die jedoch noch nicht bedient werden kann, da der Interrupt noch nicht freigegeben ist. Sobald die Freigabe erfolgt ist, springt das Programm in die IR1-Routine und ermöglicht sofort neue Interrupts. Man beachte,

daß der Eingang IR1 solange an Plus bleiben muß, bis in die IR1-Routine gesprungen wird. Die IR1-Routine kann ihrerseits nur noch durch einen IR0 unterbrochen werden.

Am Ende einer jeden Routine muß der Prozessor einen End-Of-Interrupt-Befehl EOI an den 8259 ausgeben, der daraufhin das der Routine entsprechende Bit im In-Service-Register löscht als Mitteilung, daß diese Interrupt-Routine wieder frei ist.

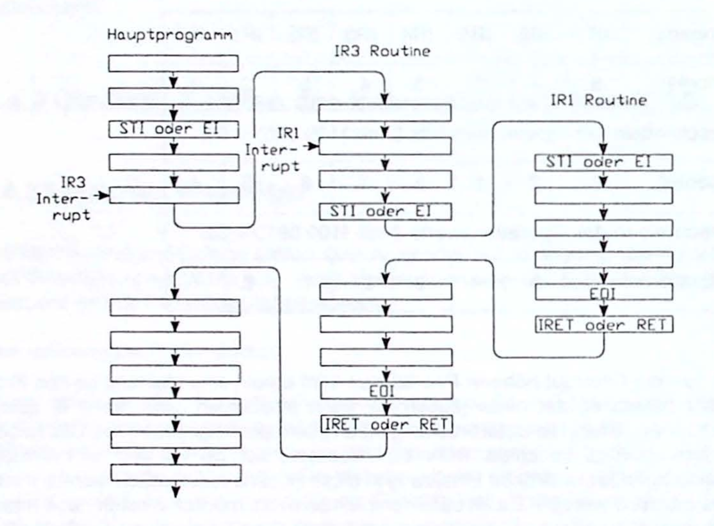


Bild 3-59. Interrupt-Verschachtelung

Einen einzelnen 8259 wird man wohl immer im vollverschachtelten Modus arbeiten lassen, es sei denn, daß gewisse Bedingungen des Programms die Reihenfolge stören. Das sind:

1. Die Benutzung des Automatic End-Of-Interrupt AEOI.
2. Die Benutzung des speziellen Maskierungsmodus.
3. Ein Slave hat einen Master, der nicht im vollverschachtelten Modus arbeitet.

Punkt 1 und 2 unterscheiden sich von Punkt 3 dadurch, daß der 8259 dem Interrupt einer niedrigeren Priorität den Vorzug vor dem einer höheren geben würde.

3.4.2.2 Automatic-End-Of-Interrupt AEOI

Durch Setzen des Bits 1 im Initialisierungswort 4 wird der 8259 in den AEOI-Modus versetzt. Der Baustein produziert in diesem Modus bei der positiven Flanke des letzten $\overline{\text{INTA}}$-Impulses, der von der CPU ausgegeben wird, seinen eigenen End-Of-Interrupt-Befehl.

Das hat Vor- und Nachteile. Die Verwendung dieses Modus befreit die Software von der Ausgabe des End-Of-Interrupt-Befehls an den 8259, hat aber den Nachteil, daß der interrupt-anfordernde Eingang bis dahin inaktiv geworden sein muß, da ansonsten erneut derselbe Interrupt aufgerufen wird. Ferner wird damit die Prioritätsstruktur des 8259 umgangen. Bei der Ausführung eines AEOI wird das Bit im In-Service-Register gelöscht, das zuvor gesetzt wurde, um anzuzeigen, welcher Interrupt gerade in Bedienung ist. Wenn nämlich weitere Interrupts von der CPU nicht ausgeschlossen worden sind, ist es möglich, daß Interrupts einer niedrigeren oder der gleichen Ebene auftreten. Wird dies nicht gewünscht, kann man entweder auf den AEOI-Modus verzichten oder die CPU akzeptiert für die Zeit der Interrupt-Bedienung keine weiteren Interrupts.

3.4.2.3 End-Of-Interrupt EOI

Wünscht man den AEOI nicht, so bleibt keine andere Wahl, als sich für den EOI-Modus zu entscheiden (Operationswort 2). Sobald ein Interrupt erkannt und von der CPU quittiert ist, wird das entsprechende Bit im In-Service-Register gesetzt. Im AEOI-Modus wird dieses Bit nahezu augenblicklich wieder gelöscht. Im EOI-Modus hingegen muß diese Aufgabe die Software erfüllen. Der Befehl muß nicht notwendigerweise am Ende der Interrupt-Routine ausgehen, sondern kann immer dann erfolgen, wenn die CPU willens ist, Interrupts niedriger Priorität zu akzeptieren. Es ist dabei möglich, entweder automatisch das zuletzt gesetzte Bit im ISR-Register zu löschen oder eines, dessen Nummer im EOI-Befehl genannt wird. Welche Art des EOI-Befehls der 8259 erkennt, hängt von der Programmierung des Operationsworts 2 ab. Man unterscheidet den nicht-spezifischen EOI-Befehl und den spezifischen EOI-Befehl.

Der nicht-spezifische EOI-Befehl:

Ist der Baustein 8259 in diesen Modus versetzt, löscht er bei Eintreffen des Befehls automatisch das zuletzt gesetzte In-Service-Bit. Um die Vorteile des nicht-spezifischen EOI-Befehls zu nutzen, muß sich der 8259 in einem Modus befinden, in dem er die In-Service-Pegel vorherbestimmen kann. Das ist im allgemeinen der spezial vollverschachtelte Modus. Aus diesem Grund sollte der nicht-spezifische EOI-Befehl nur dann benutzt werden, wenn die aktuellste und immer nur die höchste Priorität bearbeitet werden soll. Geht nämlich ein nicht-spezifischer EOI-Befehl am 8259 ein, löscht er einfach das ISR-Bit mit der höchsten Priorität, um anzuzeigen, daß die Routine mit der höchsten Priorität nun beendet ist. Auf keinen Fall ist der Befehl zu verwenden, wenn mehrere Interrupts in Bearbeitung sind und eine Rotation der Prioritäten erfolgte.

Der spezifische EOI-Befehl:

Wenn der Prozessor einen spezifischen EOI-Befehl an den 8259 sendet, läßt er ihn damit wissen, welches Bit im In-Service-Register gelöscht werden soll. Im Befehl ist also die Nummer des zu löschenden Bits enthalten (siehe Format des Operationsworts 2).

Ein spezifischer EOI-Befehl muß immer dann verwendet werden, wenn sich zwischenzeitlich die Priorität der Interrupt-Pegel, z.B. durch eine Rotation, geändert hat. Die Verwendung des nicht-spezifischen EOI-Befehls könnte in diesem Falle das falsche ISR-Bit löschen. Somit ist der spezifische EOI-Befehl immer die bessere Lösung, wenn Unklarheit herrschen sollte über die aktuellen Interrupt-Prioritäten.

3.4.2.4 Das Abschalten von Interrupt-Prioritäten

In der Regel haben die Eingänge IR eine bestimmte Prioritätsreihenfolge. Doch manchmal wünscht man eine Gleichbehandlung aller Eingänge. Das Abschalten der Prioritäten ist vor allem in solchen Anwendungen ratsam, wenn die Peripheriebausteine gleiche Dringlichkeiten aufweisen, wie es häufig bei Kommunikationskanälen der Fall ist, und geschieht durch automatische Rotation der Prioritäten. Die Philosophie des Modus ist es, zunächst alle anderen Bausteine gleicher Dringlichkeit zu bedienen, bevor erneut der ursprüngliche an der Reihe ist. Das wird dadurch erreicht, daß nach Beendigung des Interrupts der ihn auslösende Baustein mit der niedrigsten Priorität versehen wird. Schlimmstenfalls muß der Baustein warten, bis alle anderen einmal an der Reihe waren. Die Folge davon ist, daß alle Bausteine gleiche Priorität genießen und keiner bevorzugt wird.

Es gibt zwei Methoden, mit denen gleiche Prioritäten erreicht werden können: Der eine wird in Verbindung mit dem nicht-spezifischen EOI-Befehl benutzt, der andere mit dem Automatic-EOI-Modus.

Rotation beim nicht-spezifischen EOI-Befehl:

Dieser Modus wird im Operationswort 2 mit der Bitfolge 1010 0XXX eingestellt. Die Ausführung der Prioritätsrotation erfolgt erst nach Eingang eines nicht-spezifischen EOI-Befehls. Dabei wird das ISR-Bit der höchsten Priorität gelöscht und der entsprechende Interrupt-Pegel mit der niedrigsten Präferenz versehen. Die restlichen Interrutlevels rotieren in entsprechender Weise, so daß die vollverschachtelte Struktur erhalten bleibt. Die Bilder 3-60 A und B zeigen die Veränderung der Prioritätenreihenfolge nach einem nicht-spezifischen EOI-Befehl. Angenommen Interrupt 0 besitzt die höchste und Interrupt 7 die niedrigste Priorität; ferner seien das ISR3-, ISR4- und ISR6-Bit gesetzt, was heißt, daß die entsprechenden Routinen verschachtelt und gerade in Bearbeitung sind. Geht nun aus der Interrupt-Routine 3 ein EOI-Befehl im 8259 ein, wird das Bit ISR3 gelöscht und mit der Prioritätsstufe 7 versehen. ISR-Bit 3 genießt nun die höchste Priorität 0 (Bild 3-60 B).

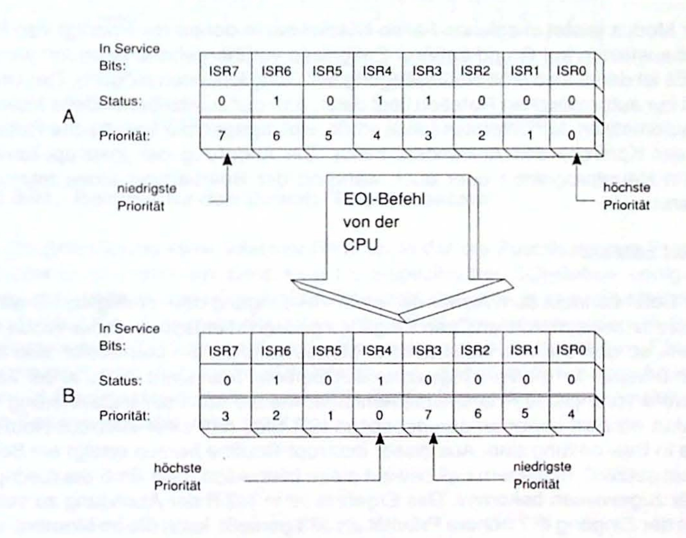


Bild 3-60. Rotation der Interrupt-Prioritäten

Rotation im Automatic-EOI-Modus:

Die Rotation im Automatic-EOI-Modus arbeitet in ganz ähnlicher Weise wie die Rotation beim Nicht spezifischen EOI-Befehl. Der wesentliche Unterschied besteht darin, daß die Prioritätsrotation automatisch nach dem letzten $\overline{INTA}$-Impuls erfolgt. Mittels Bit 7 im Operationswort 2 kann man diesen Modus wählen oder verlassen.

3.4.2.5 Spezifische Prioritätsrotation

Durch die Ausgabe eines entsprechenden Bitmusters an das Operationswort 2 des 8259 kann die Prioritätsstruktur dynamisch geändert werden. Dem Anwender stehen zwei Befehle zur spezifischen Prioritätsrotation zur Verfügung: "Priorität setzen" und "Prioritätsrotation beim spezifischen EOI-Befehl". In diesen Befehlen steckt in den drei untersten Bits die Nummer des Interrupt-Eingangs, der nachfolgend die **niedrigste** Priorität erhalten soll; daher der Name "spezifisch", da das zu setzende Bit spezifiziert werden muß. Das Bitmuster hat folgende Form: 1100 0nnn für das Setzen des niedrigsten Prioritäts-Bits und 1110 0nnn für die Rotation beim spezifischen EOI-Befehl (nnn = Nummer des Prioritäts-Bits).

Dieser Modus leistet in solchen Fällen Nützliches, in denen die Priorität von Peripheriebausteinen auf Grund äußerer Ereignisse vorübergehend geändert werden muß. Es ist damit also eine Neufestlegung der Dringlichkeiten möglich. Der Unterschied zur automatischen Rotation liegt darin, daß der zuletzt behandelte Interrupt nicht automatisch den untersten Level erhält, wohingegen die spezifische Rotation unter der Kontrolle des Anwenders bleibt. Die Änderung der Interrupt-Ebenen kann im Hauptprogramm oder auch während der Bearbeitung eines Interrupts erfolgen.

Priorität setzen:

Dieser Befehl erlaubt dem Anwender, einem IR-Eingang den niedrigsten Grad der Priorität zuzuweisen. Alle anderen Eingänge werden in entsprechender Weise verschoben, so daß der vollverschachtelte Modus aufrecht erhalten bleibt. Bild 3-61 gibt ein Beispiel für die Wirkungsweise des Befehls. Man sieht in Teil A der Abbildung die ursprüngliche Prioritätenreihenfolge, wie sie nach der Initialisierung vorliegt. Man erkennt ferner an den gesetzten ISR-Bits, daß zwei Interrupt-Routinen gerade in Bearbeitung sind. Aus dieser Interrupt-Routine heraus erfolgt ein Befehl "Priorität setzen". In diesem Fall bewirkt diese Instruktion, daß Bit 5 die niedrigste Priorität zugewiesen bekommt. Das Ergebnis ist in Teil B der Abbildung zu sehen. Da nun der Eingang IR7 höhere Priorität als IR3 genießt, kann die im Moment noch laufende Interrupt-Routine 3 beispielsweise durch einen Interrupt 6 unterbrochen

werden. Der noch anstehende Interrupt 7 muß allerdings noch warten, bis Interrupt 3 durch einen EOI-Befehl abgeschlossen ist, da die Prioritäten nur bei eintreffenden Anforderungen geprüft werden.

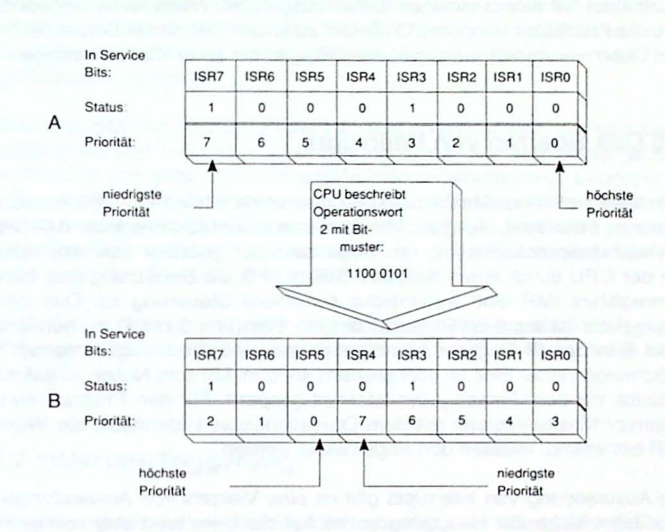


Bild 3-61. Beispiel für den Befehl: "Priorität setzen"

Bei der Beendigung einer Interrupt-Routine, in der die Zuordnung der Prioritäten geändert wurde, darf am Ende kein nicht-spezifischer EOI-Befehl erfolgen, da dieser automatisch das höchste ISR-Bit löscht und sich ja nun die Reihenfolge geändert hat. Es ist vielmehr der spezifische EOI-Befehl zu verwenden, in dem das zu löschende ISR-Bit genannt wird. Am geeignetsten ist in solchen Fällen die Verwendung des Automatic-EOI-Befehls, da dieser sofort nach Aufruf der Interrupt-Routine das entsprechende ISR-Bit löscht und dadurch keine Verwirrung im In-Service-Register verursacht.

Prioritätsrotation beim spezifischen EOI-Befehl:

Diesen Befehl kann man auffassen als eine Kombination der Befehle "spezifischer EOI" und "Priorität setzen". Daher steckt in ihm die Nummer des Bits, das nachfolgend die niedrigste Priorität besitzen und gelöscht werden soll. Es werden also zwei Aufgaben mit einem einzigen Befehl ausgeführt. Wenn keine Notwendigkeit besteht, die Prioritäten vor dem EOI-Befehl zu ändern, ist dieser Befehl der vorteilhaftere. Denn warum soll man nicht zwei Fliegen mit einer Klappe schlagen ?

3.4.2.6 Das Sperren von Interrupts

Die Hersteller reden von Maskierung und meinen damit das Ein- und Ausschalten von internen Interrupt-Leitungen. Mit dem Interrupt-Maskenregister IMR werden diese Verbindungen kontrolliert. Im Gegensatz zur globalen Interrupt-Verhinderung in der CPU durch einen Software-Befehl, läßt die Benutzung des Interrupt-Maskenregisters IMR eine wesentliche sensiblere Steuerung zu. Das Interrupt Maskenregister ist ein 8-Bit-Register, dessen Bits (von 0 bis 7) zu den entsprechenden Bits der IR-Register korrespondieren. Jeder beliebige Interrupt kann durch Schreiben einer Eins im IMR gesperrt werden. Mit acht Nullen ist jeder Interrupt erlaubt, mit acht Einsen jeder Interrupt gesperrt. Für den Programmierer ist das Interrupt Maskenregister mit dem Operationswort 1 identisch; die Wünsche, das IMR betreffend, müssen dort angemeldet werden.

Für die Ausblendung von Interrupts gibt es eine Vielzahl von Anwendungen. So können Teilbereiche der Hauptprogramms nur die Unterbrechung von gewissen Interrupts wünschen oder den Ausschluß von höheren Prioritäten für eine bestimmte Zeit. Der Beispiele gibt es viele.

Ist nun ein IMR-Bit gesetzt und erscheint am IR-Pin eine Interrupt-Anforderung, so ist sie nicht notwendigerweise vergessen, denn das IMR-Bit hat seine Wirkung nur auf den Ausgang des IRR-Flip-Flops. Auch mit einem maskierten IR-Eingang ist es noch möglich, das IRR-Flip-Flop zu setzen. Wird also eine gesperrte Leitung wieder freigegeben, hat das zur Folge, daß der entsprechende Interrupt ausgelöst wird, vorausgesetzt es handelt sich um die höchste Priorität und der High-Pegel liegt noch am IR-Eingang an. Wird dieses Signal vor der Freigabe inaktiv (Low-Pegel), wird kein Interrupt ausgelöst.

Der Spezial-Maskierungsmodus:

Manchmal haben Programmteile verstärkt Bedarf an Informationen von Bausteinen, die in der Interrupt-Hierarchie unter dem augenblicklichen Pegel liegen und somit nicht zu Wort kommen können. Man müßte mit einem Prioritätsbefehl die gesamte Struktur ändern oder mit einem EOI-Befehl das ISR-Bit löschen. Dann besteht aber die Gefahr, daß derselbe Interrupt nochmals ausgelöst werden kann und eine Überschachtelung eintritt. Wo das nicht wünschenswert ist, hilft der Spezial-Maskierungsmodus.

Dieser Modus gibt alle Interrupt-Pegel frei, ausgenommen den, der sich gerade in Bearbeitung befindet. Um das zu erreichen, setze man das IMR-Bit der gerade laufenden Routine und gebe einen Spezial-Maskierungsbefehl aus. Letzterer erfolgt durch Beschreiben von Operationswort 3 mit dem Bitmuster 0110 1000 = 68_h. Ist der Modus einmal gewählt, bleibt es solange aktiv, bis er per Befehl ausdrücklich gelöscht wird. Das Verlassen des Modus erfolgt durch Beschreiben von Operationswort 3 mit dem Bitmuster 0100 1000 = 48_h.

Dieser Modus ist nicht nur auf einen Interrupt anwendbar, sondern auch gleichzeitig auf mehrere: Es sind alle Interrupts erlaubt mit Ausnahme derer, deren IMR-Bits gesetzt sind.

3.4.2.7 Interrupt-Triggerung

Als Anwender kann man bei der Initialisierung unter zwei Interrupt-Eingabemethoden wählen, der pegel- oder der flankengesteuerten Modus, der mit Bit 3 des Initialisierungsworts 1 ausgewählt wird. Eine Eins bedeutet pegel-, eine Null flankengesteuerter Modus.

Der pegelgesteuerte Modus:

Jeder High-Pegel an einem IR-Eingang erzeugt einen Interrupt. Bleibt dieser Eingang an Plus, nachdem eine EOI-Befehl erfolgte, wird erneut derselbe Interrupt ausgelöst, wenn die INT-Leitung der CPU nicht gesperrt ist. Wenn diese Wiederholung der Interrupt-Erzeugung nicht gewünscht ist, muß man Sorge tragen, daß der IR-Eingang noch vor Ausgabe eines EOI-Befehl an Masse geht. Andererseits müssen Mindestzeiten für den High-Pegel eingehalten werden, damit eine Interrupt-Anforderung auch vom 8259 erkannt wird. Der High-Pegel muß solange anliegen, bis der erste $\overline{INTA}$-Impuls von der CPU ausgegeben wird, da der 8259 diesen Impuls als internen Takt zur Speicherung der Anforderung in die Flip-Flops benötigt. Wird ein Eingang vor dem ersten $\overline{INTA}$-Impuls an Masse gegeben, geht die Information über die genaue Herkunft des Interrupts verloren und ein Interrupt der

niedrigsten Priorität IR7 wird an die CPU gemeldet. Man sollte die dem Interrupt 7 zugeordnete Interrupt-Routine in einem System, in dem diese Möglichkeit besteht, so gestalten, daß sie den Status des 8259 abfragt oder einfach nur zur Stelle vor dem Interrupt zurückspringt.

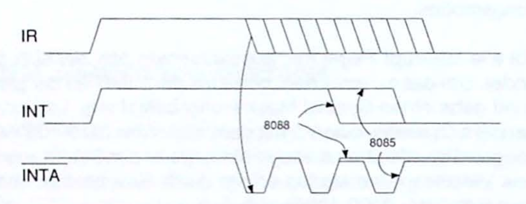


Bild 3-62. Signalformen bei der Pegel-Triggerung

Abhängig von der speziellen Konfiguration und Anwendung gibt es zahlreiche Systeme, die diesen Modus benutzen. Beispielsweise gibt es Anwendungen, in denen wiederholte Interrupt-Erzeugung notwendig ist, um ständig nach Beendigung der Interrupt-Routine erneut dieselbe Routine aufzurufen. Eine andere Anwendung ergibt sich, wenn mehrere Peripheriebausteine sich über eine Oder-Logik einen IR-Eingang teilen. Allerdings kann in einem solchen Fall die CPU nicht mehr erkennen, von welchem Baustein speziell die Anforderung stammt, und muß ihn durch gesonderte Abfrage feststellen.

Der flankengesteuerte Modus:

In diesem Modus löst nur ein Spannungswechsel von Minus nach Plus am IR-Eingang einen Interrupt aus. Der Vorteil ist darin zu sehen, daß nach Interrupt-Erkennung und Interrupt-Ende der IR-Pegel weiterhin high sein kann, ohne einen weiteren Interrupt auszulösen. Als Anwender braucht man den zeitlichen Verlauf nicht so genau im Auge zu behalten als bei der Pegelsteuerung. Allerdings muß der Pegel wieder an Masse, um erneut einen Interrupt zu erzeugen. Trotz der Flankentriggerung muß der IR-Pegel die gleiche Mindestzeit wie im pegelgesteuerten Modus high bleiben (Bild 3-62), da sonst der Standard-Interrupt 7 an die CPU gemeldet wird. Mit dieser Maßnahme kann die CPU Störungen von echten Signalen unterscheiden. Aus diesem Grund ist es empfehlenswert, ein Signal zu verwen-

den, das inaktiv High-Pegel führt und nur bei der Interrupt-Triggerung einen kurzen Masseimpuls ausgibt. Hat man einen Baustein, der zur Interrupt-Erzeugung nur einen kurzen positiven Impuls ausgibt, ist die Verwendung eines Inverters ratsam.

Im allgemeinen wird der flankengesteuerte Modus der häufiger verwendete sein, da er sich gut mit dem Automatic-End-Of-Interrupt-Modus kombinieren läßt, weniger Sorgfalt bei der zeitlichen Beobachtung des Systems erfordert und somit den Einsatz des 8259 wesentlich erleichtert.

3.4.2.8 Der Interrupt-Status

Der Zustand der internen Flip-Flops ist kein Buch mit sieben Siegeln, sondern dem Informationsdrang der CPU frei zugänglich. Jedes der 8-Bit-Interrupt-Register (Interrupt-Anforderungsregister IRR, Interrupt-Maskenregister IMR und In-Service-Register ISR) kann gelesen werden, um die Arbeit der CPU dem aktuellen Status des 8259 anzupassen.

Um den Inhalt des Interrupt-Anforderungsregisters IRR oder den des In-Service-Registers ISR zu lesen, muß die CPU zunächst den 8259 mitteilen, welches der beiden Register sie beim nächsten Lesebefehl lesen will. Das geschieht durch das Beschreiben des Operationsworts 3. Die Bits 0 und 1 treffen dabei die Auswahl:

0000 1010 : Interrupt-Anforderungsregister IRR
0000 1011 : In-Service-Register ISR

Ein normaler Lesezugriff mit Pin $A_0 = 0$ gibt nachfolgend den Inhalt des gewünschten Registers auf den Bus. Für nachfolgende Lesezugriffe auf dasselbe Register muß nicht erneut das Operationswort 3 beschrieben werden, da das Bitmuster darin gespeichert bleibt (Näheres über den Lesevorgang findet sich im Abschnitt "Programmierung"). Nach der Initialisierung ist die Grundeinstellung für das Auslesen des IRR-Registers.

Um Störungen zwischen dem Beschreiben des Operationsworts 3 und dem Auslesen des gewünschten Registers durch zwischenzeitlich erscheinende Interrupts zu vermeiden, ist es ratsam, vorübergehend die Interrupt-Unterbrechung zu sperren.

Das Lesen des Interrupt-Maskenregisters unterscheidet sich von dem Zugriff auf das IRR- oder ISR-Register dadurch, daß zuvor kein Operationswort beschrieben werden braucht. Die Unterscheidung erfolgt durch den Pin A_0. Beim Lesen von IRR oder ISR muß A_0 Low-Pegel, beim Lesen oder Beschreiben des Interrupt-Maskenregisters IMR muß A_0 High-Pegel führen. Da A_0 in der Regel mit dem Adreßbus verbunden ist, heißt das, IRR und ISR liegen an der unteren, IMR an der höheren Adresse.

3.4.2.9 Der Testbefehl

Der Automatismus der Interrupt-Erzeugung, die Beantwortung von der CPU und die Ausgabe des Sprungvektors durch den 8259 kann umgangen werden. Wenngleich in den meisten Fällen die Interrupt-Erzeugung die einfachste, schnellste und bequemste Art ist, Bedienungsanforderungen von Peripheriebausteinen über den Interruptcontroller bei der CPU anzumelden, gibt es doch Fälle, in denen diese Möglichkeit unterbunden ist. Der wichtigste Fall ist die Verwendung des 8259 in einem System, das nicht den 8080/85 oder den 8088 und seine Homologen als Zentralprozessor hat, sondern Mikrocontroller, wie z.B. den 8051 oder 8048. Diese Prozessoren, die über keine $\overline{\text{INTA}}$-Sequenz verfügen, können sich mit dieser Option trotzdem der Vorteile des Interruptcontrollers bedienen.

Mit acht kaskadierten 8259-Bausteinen sind insgesamt 64 Interrupt-Ebenen zu verwirklichen. Sollte diese Zahl insbesondere bei Kontrollanwendungen nicht ausreichen, ist bei der Verwendung des Testbefehls die Zahl erhöhbar. Die Grenze nach oben wird in einem solchen System von der Zahl der Bausteine bestimmt, die der Prozessor adressieren kann.

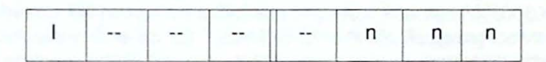
Ein anderer Aspekt, bei dem diese Testmethode von Vorteil sein kann, sind zeitkritische Programmteile, die zwar Informationen von der Peripherie benötigen, die aber nicht an unerwünschter Stelle durch einen Interrupt unterbrochen werden dürfen. In beiden Fällen läßt sich der 8259 auf eingegangene Interrupt-Anforderungen testen. Der Vorteil ist darin zu sehen, daß der Prozessor nicht selbst jeden Peripheriebaustein adressieren muß, sondern lediglich den 8259 zu fragen braucht, und kann dabei die Vorteile genießen, die die verschiedenen Prioritätsmodi und Interrupt-Verwaltung zur Verfügung stellt. Die Prozessoren 8080/85 und 8088 und Homologe können den Testbefehl neben dem Interrupt-Modus in demselben System benutzen, müssen aber bei der Verwendung des Testbefehls den Interrupt-Eingang INT durch die Software sperren.

Der Testbefehl an den 8259 wird durch Beschreiben des Operationsworts 3 ausgeführt, wenn dabei Bit 2 gesetzt wird. Eine automatische Wiederholung ist nicht möglich. Vor jeder Abfrage muß erneut dieses Bit im Operationswort 3 gesetzt werden.

Bitmuster des Testbefehls: 0000 1100 = 0C

Sobald dieses Bit gesetzt ist, betrachtet der 8259 den folgenden Lesezugriff als ein Interrupt-Anerkennungssignal und gibt ein Byte auf den Bus, das anzeigt, ob ein Interrupt anliegt, und das die Nummer des Interrupts mit der höchsten Priorität enthält. Der Prozessor kann daraus seine eigenen Schlüsse ziehen und in entsprechende Subroutinen springen.

Format des Test-Bytes:



I	--	--	--	--	n	n	n

Dabei geben die Bits nnn die binärkodierte Nummer des Interrupts mit der höchsten Priorität an. Das Bit I zeigt mit einer Eins an, daß ein Interrupt vorliegt, und mit einer Null das Fehlen einer Anforderung. Im letzten Fall weisen die drei Bits nnn Einsen auf.

3.4.2.10 Die Interrupt-Kaskadierung

Mit den Anschlüssen CAS_{0-2} und den Eingängen IR_{0-7} ist eine Erweiterung der acht Interrupt-Ebenen auf 64 möglich. Dabei sind die Pins CAS_{0-7} bei einem Slave als Eingänge und beim Master als Ausgänge geschaltet. Sie werden miteinander verbunden und bilden einen gesonderten 3-Bit-Adreßbus. Der Ausgang INT der Slaves ist nicht an den Prozessor geführt, sondern an einen IR-Eingang des Masters, der den Interrupt aus zweiter Hand an den Prozessor weiterleitet (Bild 3-63). Für die Zusammenarbeit von Master mit Slave sind die Bausteine als solche programmierbar, damit keine Verwirrung über die Zuständigkeiten entsteht.

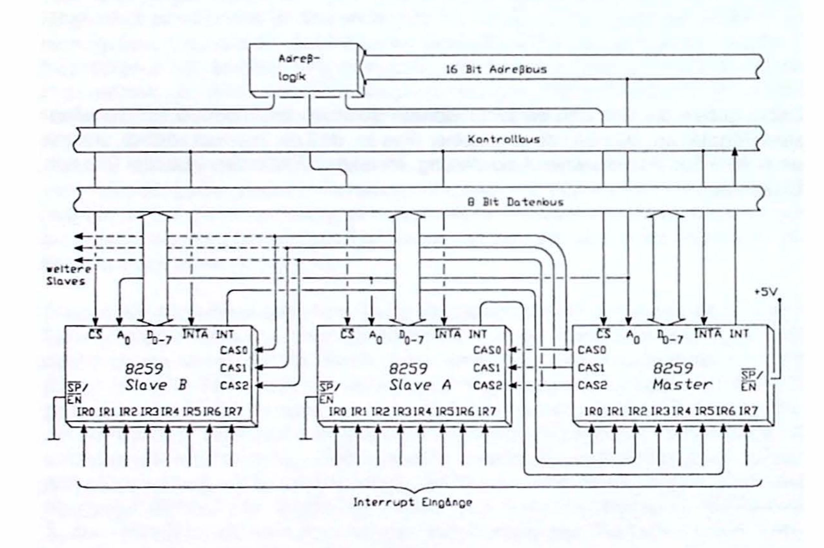


Bild 3-63. Die Kaskadierung von 8259-Interruptcontrollern

Master - Slave:

In einem System mit mehreren 8259 muß einer zum Master gewählt werden, der die Kontrolle über die Slaves ausübt. Die Unterscheidung zwischen Master und Slave kann auf zwei Arten erfolgen.

1. Hardware: Der Pin $\overline{SP}/\overline{EN}$ des Masters wird an High-Pegel, der der Slaves an Masse gelegt.

2. Software: Durch Beschreiben der Bits 2 und 3 im Initialisierungswort 4 wird die Unterscheidung vorgenommen mit

    000X 11XX für den Master
    und 000X 10XX für den Slave.

Wird die Unterscheidung durch die Hardware vorgenommen, muß das entsprechende Bitmuster im Initialisierungswort 4 lauten: 000X 0XXX, da durch das Setzen einer Eins im Bit 3 der Pin $\overline{SP}/\overline{EN}$ als Ausgang definiert wird und für andere Verwendung im System vorgesehen ist (—> gepufferter Modus). In diesem Falle ist der Ausgang $\overline{SP}/\overline{EN}$ freizulassen.

Der Ausgang INT des Slave wird mit einem IR-Eingang des Masters verbunden. Dabei spielt die Reihenfolge der Anschlüsse im Prinzip keine Rolle. Nicht für Slaves verwendete IR-Eingänge des Masters stehen weiterhin für Peripheriebausteine zur Verfügung. Für eine funktionierende Zusammenarbeit zwischen Master und Slave muß der Master wissen, welcher Slave an welchem IR-Eingang angeschlossen ist, und andererseits muß der Slave seine eigene Nummer kennen, damit er durch den Master adressiert werden kann. Diese Mitteilung erfolgt für den Master und für den Slave im Initialisierungswort 3. Der Master erfährt dadurch, an welchen Eingängen Slaves angeschlossen sind; der Slave erfährt seine binärkodierte Nummer, die er im Falle einer Kommunikation mit dem Bitmuster auf den Leitungen CAS_{0-2} vergleicht.

Trifft ein Interrupt bei einem Slave ein, durchläuft er die Interrupt-Kontrolle und wird im Falle seiner Anerkennung über die Leitung INT beim Master angemeldet. Dieser behandelt das Signal wie eines der üblichen eingehenden Interrupts und löst nach Kontrolle bei der CPU einen Interrupt aus. Der Prozessor reagiert in der gleichen Weise wie in einem nicht erweiterten System und gibt den $\overline{INTA}$-Antwortimpuls aus, der sowohl Master als auch Slave erreicht. Während die CPU von der Master-Slave-Struktur keine Notiz nimmt, ist das nachfolgende Verhalten der 8259er Bausteine unterschiedlich. Der erste $\overline{INTA}$-Impuls wird von allen 8259ern für interne Taktzwecke benutzt, und im 8085-Modus gibt der Master gleichzeitig den CALL-Befehl auf den Datenbus. Ferner bewirkt der erste $\overline{INTA}$-Impuls die Ausgabe der Slave-Adresse über die Leitungen CAS_{0-2} durch den Master. Damit ist

die Kontrolle an den Slave übergeben und die Aufgabe des Masters für diesen Interrupt-Zyklus beendet. Mit dem nachfolgenden $\overline{INTA}$-Impuls (zwei im 8085-Modus) gibt nun der Slave die Vektoradresse über den Bus an die CPU aus.

In der Interrupt-Routine muß sich der Prozessor daran erinnern, daß er bei der Initialisierung die Master-Slave-Struktur programmiert hat und muß sowohl an den Master als auch an den Slave einen End-Of-Interrupt-Befehl EOI ausgeben, da in beiden Bausteinen das In Service-Bit gesetzt wurde (vorausgesetzt, die Bausteine befinden sich nicht im Automatic-End-Of-Interrupt AEOI).

Eine gewisse Aufmerksamkeit ist der Hardware zu schenken, wenn nicht jeder IR-Eingang des Masters mit einem Slave besetzt ist und wenn sowohl Slaves als auch Peripheriebausteine einen Interrupt auslösen. In diesem Falle darf der Eingang IR0 nicht von einem Slave belegt sein. Das hat seinen Grund darin, daß die Leitungen CAS_{0-2} im inaktiven Zustand, wenn also ein Peripheriebaustein einen Interrupt erzeugt, Low-Pegel führen. Wäre ein Slave an IR0 angeschlossen, würde mit jedem Nicht-Slave-Interrupt der Slave aktiviert. Somit ist bei Belegung der IR-Eingänge des Master mit Slaves der Eingang IR0 der letzte, der belegt werden darf.

3.4.2.11 Der spezial-vollverschachtelte Modus

Abhängig von der betreffenden Anwendung sind manchmal Änderungen der verschachtelten Struktur im kaskadierten Modus wünschenswert, da die verschachtelte Struktur eines Slaves sich vom normalen vollverschachtelten Modus unterscheidet. Geht nämlich bei einem Slave während einer Interrupt-Routine, die derselbe Slave ausgelöst hat, eine Interrupt-Anforderung höherer Priorität ein, müßte der Slave über denselben IR-Eingang beim Master erneut einen Interrupt auslösen, der aber durch das gesetzte In-Service-Bit gesperrt ist. Somit könnte nur ein anderer Slave von höherer Priorität die laufende Interrupt-Routine unterbrechen.

Mit dem spezial-vollverschachtelten Modus ist es möglich, daß der Master alle Interrupts niedrigerer Priorität als das gesetzte ISR-Bit ignoriert und sich nur durch Interrupts gleicher oder höherer Priorität unterbrechen läßt. Der spezial-vollverschachtelte Modus kann nur im Master bei der Initialisierung programmiert werden.

Für eine geordnete Arbeitsweise im spezial-vollverschachtelten Modus muß die Software vor Ausgabe eines EOI-Befehls an den Master feststellen, ob noch weitere Slave-Interrupts in Bearbeitung sind. Das kann erreicht werden, indem man

an den Slave einen EOI-Befehl gibt und anschließend das In-Service-Register liest. Ist der Inhalt Null, stehen keine weiteren Interrupts mehr an und ein EOI-Befehl kann nun an den Master gegeben werden.

3.4.2.12 Der gepufferte Modus

Wie bereits erwähnt besitzt der Pin $\overline{SP}/\overline{EN}$ eine Doppelfunktion. Wenn bei der Initialisierung nicht anders programmiert, ist dieser Pin Eingang und wird in Systemen ohne Master-Slave-Struktur an High-Pegel gelegt. Durch das Anlegen von Masse wird dem Chip mitgeteilt, daß er ein Slave ist. Im gepufferten Modus, der mit Bit 3 des Initialisierungsworts 4 gewählt wird, ist der Pin $\overline{SP}/\overline{EN}$ Ausgang und wird zur Steuerung eines Datenbustreibers verwendet, der in großen Systemen mit langen Busleitungen vonnöten ist. Der Ausgang $\overline{SP}/\overline{EN}$ führt in der Regel High-Pegel und wird nur dann low-aktiv, wenn der Datenbusausgang D_{0-7} freigegeben ist.

In Bild 3-64 sind drei kaskadierte 8259er zu sehen. Jeder Slave kontrolliert einen eigenen bidirektionalen 8-Bit-Bustreiber vom Typ 8286 oder 74HC245. Zu beachten ist der Pull-Up hinter dem Ausgang $\overline{SP}/\overline{EN}$. Er dient dem Zweck, die Daten zur Initialisierung zum Interruptcontroller fließen zu lassen; schaltet also den Transceiver von B nach A. Wenn die Daten vom 8259 zum Prozessor wandern sollen, geht der Pin $\overline{SP}/\overline{EN}$ an Masse, ansonsten ist er high.

Im gepufferten Modus ist natürlich der Pin $\overline{SP}/\overline{EN}$ nicht mehr zur Unterscheidung zwischen Master und Slave zu verwenden, das muß in diesem Fall durch die Software (Bit 2, Initialisierungswort 4) erfolgen.

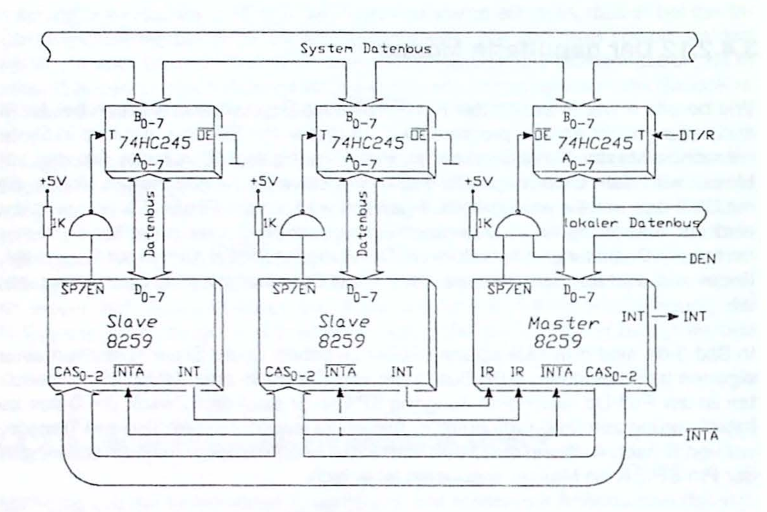


Bild 3-64. Schaltungsbeispiel für den gepufferten Modus

3.4.3 Die Programmierung des 8259

Bei der Programmierung des Bausteins ist prinzipiell zwischen zwei verschiedenen Befehlswörtern zu unterscheiden: dem Initialisierungswort (IW) und dem Operationswort (OW). Wie der Name sagt, wird das Initialisierungswort einmal in der Reset-Routine des Prozessors in den 8259 geschrieben und dient zur Anpassung des Bausteins an die individuelle Systemkonfiguration. Für eine korrekte Programmierung ist es unabdingbar, daß der Baustein zuerst mit den Initialisierungswörtern beschreiben worden ist. Beim Schreiben muß die folgende Sequenz beachtet werden:

1. Schreiben des Initialisierungsworts 1, A_0= 0
2. Schreiben des Initialisierungsworts 2, A_0= 1
3. Sind mehrere 8259 im System, schreibe das Initialisierungswort 3, A_0= 0.
4. Wenn das Bit IW4 im Initialisierungswort 1 gesetzt war, schreibe das Initialisierungswort 4, A_0= 0.

Zu beachten ist bei der Verwendung mehrerer Bausteine, daß ein jeder nach der vorstehenden Sequenz programmiert werden muß.

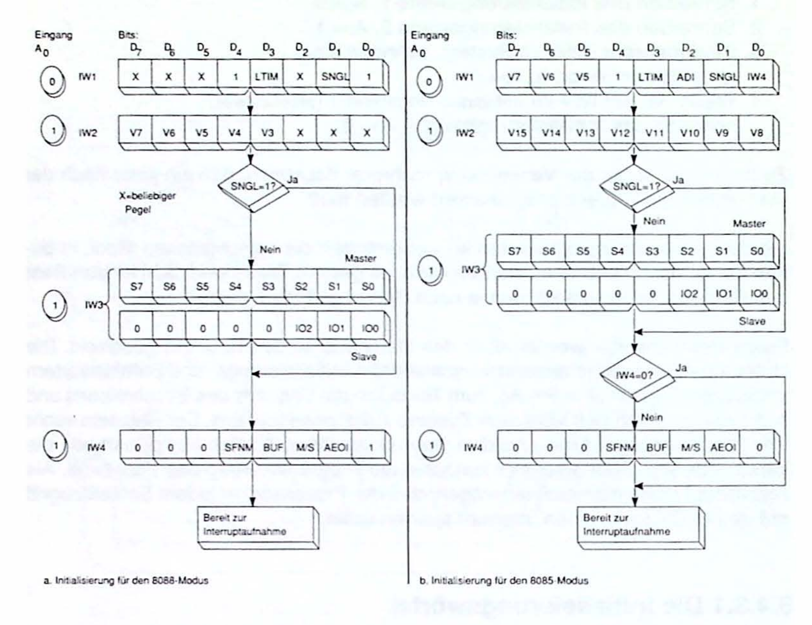
Mit dem Operationswort werden im wesentlichen die verschiedenen Modi, in denen der Baustein betrieben werden kann, eingestellt. Diese sind nicht für den Rest des Betriebs fixiert und können je nach Bedarf geändert werden.

Beide Befehlswörter werden über den Datenbus an den Baustein geschickt. Die Unterscheidung zwischen den verschiedenen Initialisierungs- und Befehlswörtern erfolgt zum Teil mit dem Pin A_0, zum Teil durch die Sequenz des Beschreibens und nicht zuletzt durch den logischen Zustand zuvor gesetzter Bits. Der Baustein kennt vier Initialisierungswörter und drei Operationswörter. Ähnlich kompliziert wie die verschiedenen Modi erscheint zunächst die Programmierung des Bausteins. Als Faustregel sollte man sich einprägen, daß der Prozessor bei jedem Schreibzugriff auf den 8259 zuvor einen Interrupt sperren sollte.

3.4.3.1 Die Initialisierungswörter

Vor einer geregelten Arbeitsweise muß jeder Interruptcontroller im System mit einer Abfolge von zwei bis vier Bytes initialisiert, d.h. programmiert werden. Diese Bytes heißen Initialisierungswörter und werden benötigt, um den 8259 in eine für das System geeignete Arbeitsweise zu versetzen. Dabei müssen nicht immer alle vier Wörter geschrieben werden. Die Anzahl hängt von gewissen Konditionen ab, die in Bild 3-65 genannt sind.

In jedem Fall müssen die Initialisierungswörter 1 und 2 geschrieben werden. Das Beschreiben der restlichen Wörter hängt von gewissen Bits des zuvor geschriebenen Initialisierungsworts 1 ab.



a. Initialisierung für den 8088-Modus

b. Initialisierung für den 8085-Modus

Bild 3-65. Initialisierungsalgorithmus des Interruptcontrollers 8259

3.4.3.1.1 Übersicht über die Initialisierungswörter

IW1: An zwei Kriterien erkennt der 8259 das erste Initialisierungswort:

1. Die Adreßleitung A_0 muß eine Null aufweisen und
2. das Bit D_4 muß eine Eins sein.

Wäre D_4 eine Null, würde das der Interruptcontroller als das Operationswort 2 oder 3 auffassen.

IW2: Dieses Initialisierungswort wird immer unmittelbar nach dem Initialisierungswort 1 ausgegeben. Um es zu adressieren, muß die Adreßleitung A_0 eine Eins aufweisen. Mit diesem Wort erhält der 8259 Informationen über die Vektoradresse, die er im Falle eines Interrupts an die CPU ausgeben wird.

IW3: Das Initialisierungswort wird nur dann benötigt, wenn das Bit SNGL im Initialisierungswort 1 mit einer Null beschrieben wurde. Um das Wort zu schreiben, muß die Adreßleitung A_0 High-Pegel führen. Es wird immer dann benötigt, wenn mehrere 8259er mit Master-Slave-Struktur im System vorhanden sind. Dem Master teilt es mit, an welchem Eingang ein Slave vorhanden ist und welche Nummer er führt; dem Slave teilt es mit, mit welcher Adresse der Master ihn anspricht.

IW4: Die Ausgabe dieses Initialisierungsworts ist abhängig vom Zustand des Bits 0 im Initialisierungswort 1. Mit einer Eins ist IW4 erforderlich. Um es zu schreiben, muß auch in diesem Falle die Adreßleitung A_0 High-Pegel führen. Es wird genau dann benötigt, wenn irgendein Bit darin einen anderen Zustand als die Null haben soll, denn beim Schreiben des Initialisierungsworts 1 werden alle Bits darin gelöscht.

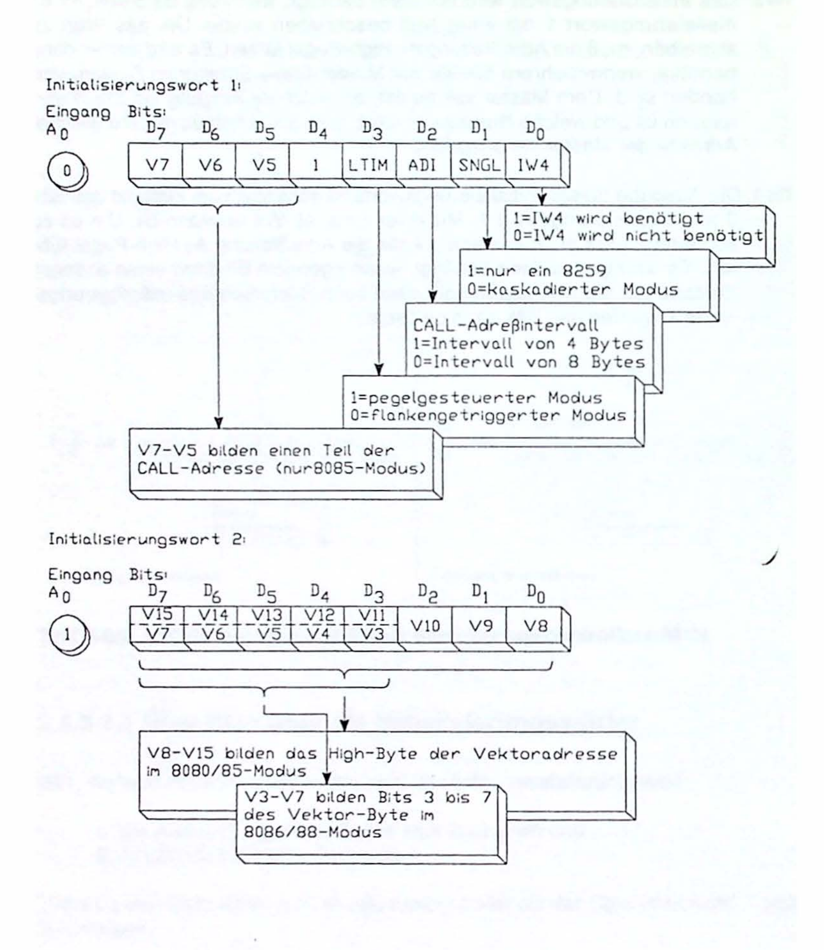


Bild 3-66. Das Format der Initialisierungswörter 1 und 2

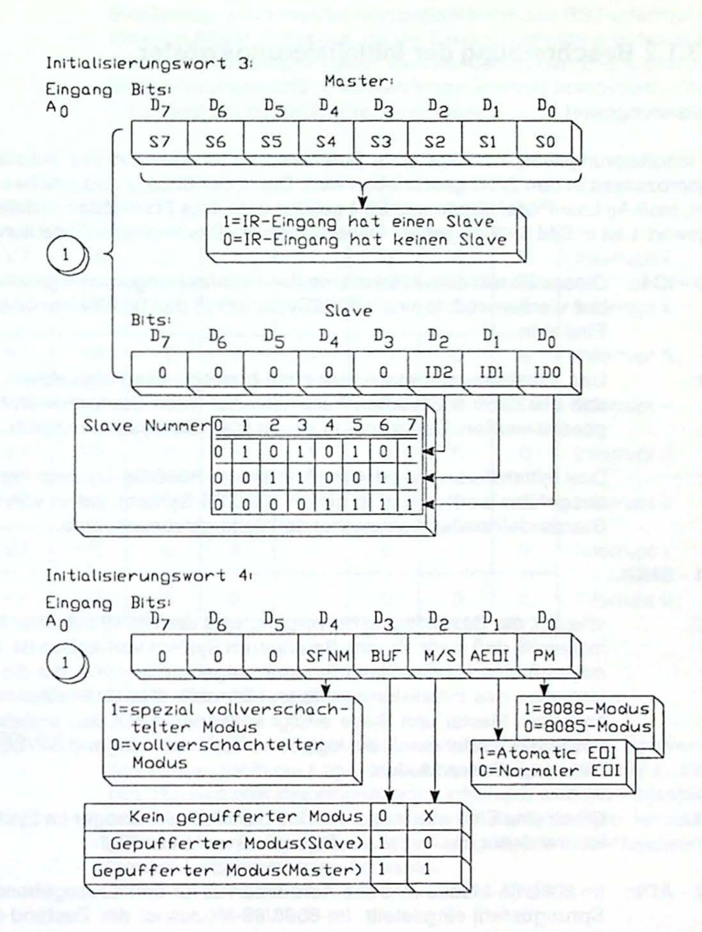


Bild 3-67. Das Format der Initialisierungswörter 3 und 4

3.4.3.1.2 Beschreibung der Initialisierungswörter

Initialisierungswort 1:

Das Initialisierungswort 1 ist das erste Befehlswort, das während des Initialisierungsprozesses in den 8259 geschrieben wird. Damit der 8259 es als solches erkennt, muß A_0 Low-Pegel führen und Bit 4 gesetzt sein. Das Format des Initialisierungswort 1 ist in Bild 3-66 zu sehen. Dabei haben die Bits folgende Bedeutung:

Bit 0 - IC4: Dieses Bit teilt dem 8259 mit, ob das Initialisierungswort 4 geschrieben werden wird. In einem 8088System muß das Bit IW4 immer eine Eins sein.

0: Das Initialisierungswort 4 wird nicht benötigt. Die Folge davon ist, daß alle darin enthaltenen Parameter auf ihren Standardwert Null gesetzt werden. Das ist nur in einem 8080/85-System möglich.

1: Das Initialisierungswort 4 muß in einem 8086/88-System immer ausgeführt werden und in einem 8080/85-System, wenn von der Standardeinstellung abweichende Werte gewünscht sind.

Bit 1 - SNGL:

0: Wie aus der Bezeichnung hervorgeht, wird dem 8259 mit einer Null mitgeteilt, daß mehr als eine Baustein im System vorhanden ist. Somit muß eine Master-Slave-Struktur aufgebaut werden, die die Information des Initialisierungsworts 3 braucht. Die Unterscheidung zwischen Master und Slave erfolgt entweder durch das Initialisierungswort 4 oder durch die logischen Pegel am Eingang $\overline{SP}/\overline{EN}$ im nicht gepufferten Modus.

1: Durch eine Eins wird mitgeteilt, daß der 8259 als einziger im System ist und daher das Initialisierungswort 3 nicht benötigt.

Bit 2 - ADI: Im 8080/85-Modus wird das *Adreßin*tervall für den auszugebenden Sprungbefehl eingestellt. Im 8086/88-Modus ist der Zustand des Bits belanglos.

0: Das vom 8259 erzeugte Adreßintervall umfaßt 8 Bytes. Diese Einstellung gewährleistet Kompatibilität mit den RST-Interrupt-Vektoren im 8080/85-System, da die Einsprungstellen 8 Bytes auseinander liegen. Dieser Vektor wird mit den in den Bits 6 und 7 des Initialisierungsworts 1 vorhandenen Werten kombiniert. Tabelle 3-13 zeigt die dadurch erzeugten Werte.

D7	D6	D5	D4	D3	D2	D1	D0	
V7	V6	1	1	1	0	0	0	Interrupt 7
V7	V6	1	1	0	0	0	0	Interrupt 6
V7	V6	1	0	1	0	0	0	Interrupt 5
V7	V6	1	0	0	0	0	0	Interrupt 4
V7	V6	0	1	1	0	0	0	Interrupt 3
V7	V6	0	1	0	0	0	0	Interrupt 2
V7	V6	0	0	1	0	0	0	Interrupt 1
V7	V6	0	0	0	0	0	0	Interrupt 0

Tabelle 3-13. 8-Byte-Adreßintervall

1: Das vom 8259 erzeugte Adreßintervall umfaßt 4 Bytes. Das versetzt den Anwender in die Lage, eine kompakte Sprungtabelle zu benutzen. Die Nummer des eingehenden Interrupts wird mit 4 multipliziert und die Bits V5 bis V7 hinzuaddiert, um das Low-Byte der Sprungadresse zu errechnen. Tabelle 3-14 zeigt die den verschiedenen Interrupts entsprechenden Adressen.

D7	D6	D5	D4	D3	D2	D1	D0	
V7	V6	V6	1	1	1	0	0	Interrupt 7
V7	V6	V6	1	1	0	0	0	Interrupt 6
V7	V6	V6	1	0	1	0	0	Interrupt 5
V7	V6	V6	1	0	0	0	0	Interrupt 4
V7	V6	V6	0	1	1	0	0	Interrupt 3
V7	V6	V6	0	1	0	0	0	Interrupt 2
V7	V6	V6	0	0	1	0	0	Interrupt 1
V7	V6	V6	0	0	0	0	0	Interrupt 0

Tabelle 3-14. 4-Byte-Adreßintervall

Bit 3 - LTIM:

0: Die Eingänge IR sind flankengetriggert. Eine Interrupt-Anforderung auf einer der IR-Leitungen wird durch einen Übergang von Low nach High ausgelöst. Das IR-Signal muß danach solange High-Pegel beibehalten, bis die negativen Flanke des ersten $\overline{\text{INTA}}$-Impulses vorbei ist.

1: Dadurch wird der pegelgetriggerte Modus gewählt. Ein Interrupt wird nun mit High-Pegel an einem IR-Eingang ausgelöst. Allerdings muß der High-Pegel wieder entfernt sein, bevor ein End Of Interrupt-Befehl (EOI) das In Service Bit (ISR) löscht, da ansonsten sofort erneut derselbe Interrupt aktiv wird.

Bit 4 - 1: In Verbindung mit dem Eingang A0=0 erkennt der 8259, daß es sich bei diesem Byte um das Initialisierungswort 1 handelt.

Bits 5 - 7: Diese Bits werden im 8080/85-Modus benutzt, um einen Teil des Low-Bytes zu bilden, das bei einer Interrupt-Beantwortung durch die CPU als Adresse an sie ausgegeben wird. Nur im 4-Byte-Adreß-intervall werden alle drei Bits gebraucht, im 8-Byte-Intervall ist der Zustand von Bit 5 bedeutungslos. Ebenso bedeutungslos sind die Zustände der Bits, wenn der 8259 in den 8088-Modus versetzt ist.

Initialisierungswort 2:

Das Initialisierungswort 2 ist das zweite Kontrollwort, das in den 8259 geschrieben werden muß. Abhängig vom gewählten System (8085 oder 8088) wird dieses Byte auf eine von zwei Arten benutzt. In Verbindung mit einem 8080/85-Prozessor stellt dieses Byte das Adressen-High-Byte des CALL-Befehls dar und wird unverändert bei der Interrupt-Antwort ausgegeben.

In einem 8086/88-System wird in die ersten drei Bits die Nummer des auslösenden Interrupts geschrieben und die restlichen Bits laut Programmierung hinzugefügt. Im Prozessor wird dieses Byte mit vier multipliziert. Aus der so errechneten Adresse holt sich die CPU die eigentliche Sprungadresse für die geforderte Interrupt-Routine. Somit müssen alle Sprungvektoren in Viererschritten an den Adressen von 0 bis 1020 stehen (Bild 3-68).

Initialisierungswort 3:

Das Initialisierungswort 3 muß nur dann beschrieben werden, wenn das Bit SNGL im Initialisierungswort 1 gelöscht ist. Ist es gesetzt, wird das nächste in den 8259 geschriebene Wort als Initialisierungswort 4 interpretiert, vorausgesetzt der Pegel von A_0=1 und das Bit IW4 im Initialisierungswort 1 waren gesetzt. Ansonsten wird es als ein Operationswort betrachtet.

Das Initialisierungswort 3 kann auf zwei Arten interpretiert werden, die davon abhängen, ob der Baustein als Master oder als Slave verwendet wird. Die Unterscheidung erfolgt entweder durch die Software im Initialisierungswort 4 oder durch die Hardware mit dem Pin $\overline{SP}/\overline{EN}$ (Pin 16).

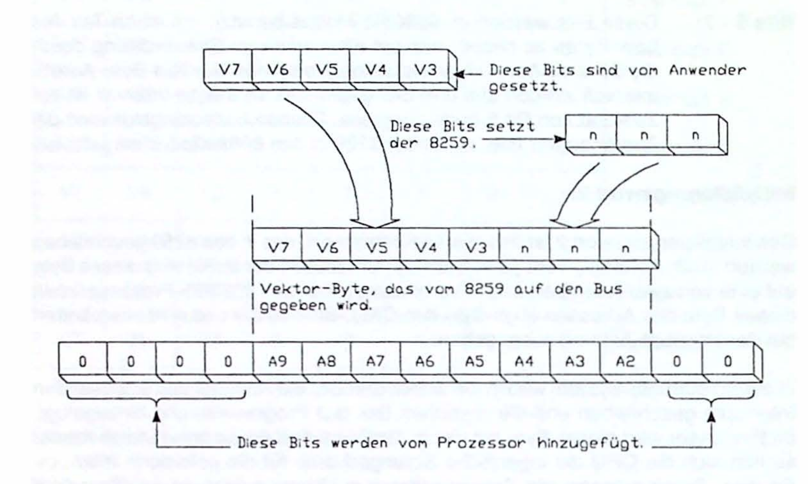


Bild 3-68. Aufbau des Interrupt-Vektors im 8088-System

Der 8259 als Master:

Im Master muß das Initialisierungswort 3 in allen Bits, wo an den entsprechenden IR-Eingängen ein Slave angeschlossen ist, eine Eins aufweisen. Eine Null in den entsprechenden Bits teilt dem 8259 mit, daß dieser Eingang entweder frei oder mit einem anderen Peripheriebaustein belegt ist. Bei der Hardware-Entwicklung soll darauf geachtet werden, daß der Eingang IR0 zuletzt mit einem Slave verbunden wird (siehe Abschnitt 3.4.2.10).

Der 8259 als Slave:

Jeder Slave benötigt die Information über seine Nummer, um an der richtigen Stelle selektiert zu werden. Die Nummer entspricht der Nummer des IR-Eingangs am Master, mit dem der Slave verbunden ist. Löst der Slave am Master einen Interrupt

aus, so weiß der Master auf Grund der Programmierung, von welchem Slave die Anforderung stammt, und bittet ihn über die Leitungen CAS_{0-2}, der CPU den Interrupt-Vektor mitzuteilen.

Um den Slave zu numerieren, werden nur die drei untersten Bits des Initialisierungsworts 3 benötigt, da die Mitteilung im Binärkode erfolgt. Die Slave-Identifikation zeigt Tabelle 3-15.

IR-Eingang am Master	IO2	IO1	IO0
IR7	1	1	1
IR6	1	1	0
IR5	1	0	1
IR4	1	0	0
IR3	0	1	1
IR2	0	1	0
IR1	0	0	1
IR0	0	0	0

Tabelle 3-15. SlaveIdentifikation durch das Initialisierungswort 3

Das Initialisierungswort 4:

Dieses Initialisierungswort wird nur dann geschrieben, wenn das Bit IW4 im Initialisierungswort 1 gesetzt wurde. Es nur dann benötigt, wenn andere Werte als die Standardwerte (alle Bits=0) benutzt werden sollen. Die Bits haben folgende Bedeutung:

Bit 0 - PM: Prozessor-Modus. Mit diesem Bit wird dem 8259 mitgeteilt, mit welchem Typ von Prozessor er verbunden ist. Das ist wesentlich, da es stark die Interrupt-Antwortsequenz beeinflußt.

0: Der 8259 wird in einem 8080/85-System verwendet.

1: Der 8259 wird in einem 8086/88-System und Verwandte verwendet.

Bit 1 - AEOI: Mit diesem Bit wird dem 8259 mitgeteilt, ob ein nicht spezifizierter Automatic-End-Of-Interrupt-Befehl ausgeführt werden soll oder nicht. Es ist dabei zu beachten, daß eine verschachtelte Prioritätsstruktur verloren gehen kann.

0: Ein Automatic-End-Of-Interrupt wird nicht ausgelöst.

1: Ein Automatic-End-Of-Interrupt wird bei der positiven Flanke des letzten $\overline{\text{INTA}}$-Impulses erzeugt.

Bit 2 - M/S: Dieses Bit zeigt nur dann eine Wirkung, wenn das Bit 3 BUF gesetzt ist. Der Zweck dieses Bits ist es, durch die Software eine Master-Slave-Struktur herzustellen.

0: Der 8259 wird als Slave benutzt.

1: Der 8259 wird als Master benutzt.

Bit 3 - BUF: Dieses Bit teilt dem 8259 mit, ob die Daten, die an den Ausgängen D_0-D7 anliegen, durch einen externen Treiber gepuffert werden. Im gepufferten Modus wird der Pin $\overline{\text{SP}}/\overline{\text{EN}}$ Ausgang und geht immer dann an Masse, wenn der Baustein Daten ausgibt. Mit diesem Signal wird die Datenrichtung in einem Transceiver bestimmt.

0: Der Baustein wird im nicht gepufferten Modus betrieben. Das hat zwei Dinge zur Folge: 1. Der Zustand des M/S-Bits ist gleichgültig. 2. Der Pin $\overline{\text{SP}}/\overline{\text{EN}}$ wird Eingang und entscheidet über Master-Slave-Funktion. Für Systeme mit nur einem Baustein muß in diesem Fall der Pin $\overline{\text{SP}}/\overline{\text{EN}}$ an Plus gehalten werde.

1: Der Baustein befindet sich im gepufferten Modus. Der Pin $\overline{\text{SP}}/\overline{\text{EN}}$ wird Ausgang und erzeugt ein Freigabesignal bei der Datenausgabe. Die Master-Slave-Struktur muß mit dem Bit M/S eingestellt werden.

Bit 4 - SFNM: Für mehrere Bausteine, die in einem System kaskadiert sind, wählt dieses Bit den spezial-vollverschachtelten Modus und darf nur im Master aktiv werden.

0: Der spezial-vollverschachtelte Modus ist nicht gewählt.

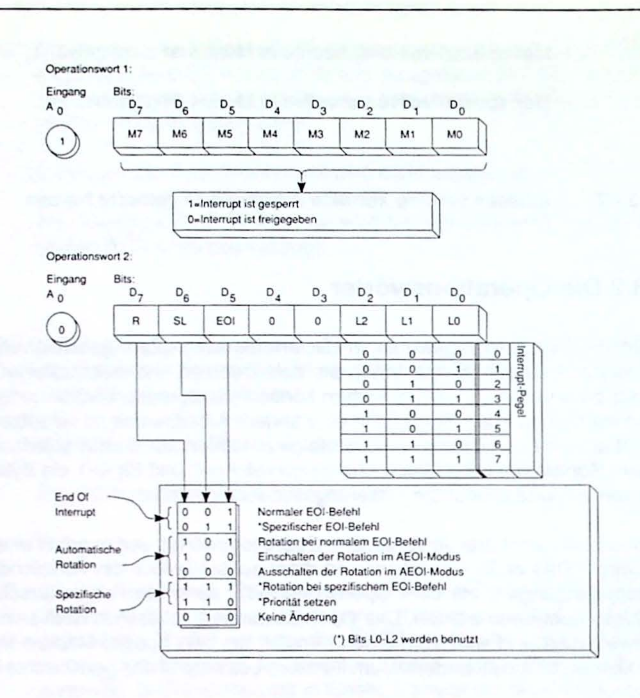
1: Der spezial-vollverschachtelte Modus ist gewählt.

Bits 5 - 7 müssen für eine korrekte Arbeitsweise gelöscht bleiben.

3.4.3.2 Die Operationswörter

Sobald die Initialisierungswörter in der erforderlichen Zahl geschrieben worden sind, ist der Baustein bereit, Interrupts aufzunehmen und weiterzuleiten. Im Gegensatz zu den Initialisierungswörtern können die Operationswörter jederzeit geändert werden, um den Baustein in eine andere Arbeitsweise zu versetzen. Dabei ist nicht auf eine vorbestimmte Reihenfolge zu achten. Ein Schritt ist jedoch zu vermeiden: Sobald die CPU fälschlicherweise mit A_0=0 und Bit 4=1 ein Byte an den 8259 sendet, wird erneut die Initiierungssequenz gestartet.

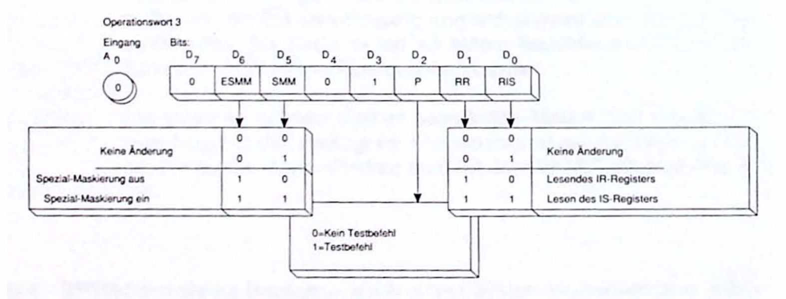
Der Baustein kennt drei verschiedene Operationswörter. Jedes erfüllt einen anderen Zweck. Das erste Operationswort dient ausschließlich der Ausblendung von Interrupt-Eingängen; mit dem Operationswort 2 kann man unter verschiedenen Prioritätsalgorithmen wählen. Das Operationswort 3 schließlich ist in seiner Funktion zweigeteilt und wird zum einen benutzt, um den Spezial-Masken-Modus zu kontrollieren, und zum anderen, um für einen Lesezugriff das gewünschte Register zu wählen.



L2	L1	L0	Interrupt-Pegel
0	0	0	0
0	0	1	1
0	1	0	2
0	1	1	3
1	0	0	4
1	0	1	5
1	1	0	6
1	1	1	7

R	SL	EOI	
0	0	1	Normaler EOI-Befehl
0	1	1	*Spezifischer EOI-Befehl
1	0	1	Rotation bei normalem EOI-Befehl
1	0	0	Einschalten der Rotation im AEOI-Modus
0	0	0	Ausschalten der Rotation im AEOI-Modus
1	1	1	*Rotation bei spezifischem EOI-Befehl
1	1	0	*Priorität setzen
0	1	0	Keine Änderung

Bild 3-69. Das Format der Operationswörter 1 und 2



	ESMM	SMM
Keine Änderung	0	0
	0	1
Spezial-Maskierung aus	1	0
Spezial-Maskierung ein	1	1

RR	RIS	
0	0	Keine Änderung
0	1	
1	0	Lesendes IR-Registers
1	1	Lesen des IS-Registers

Bild 3-70. Das Format des Operationsworts 3

Das Operationswort 1:

Auf das Operationswort 1 wird genau dann zugegriffen, wenn der Eingang A_0 High-Pegel führt und die Initiierungssequenz abgeschlossen ist. Dieses Kontrollwort dient dem Zweck, die acht Interrupt-Linien mittels Interrupt-Maskenregister ein bzw. auszuschalten. Bei der Initialisierung des Bausteins wird der Inhalt des Operationsworts 1 auf Null gesetzt, womit alle Interrupt-Eingänge freigegeben sind. Deswegen ist ein Beschreiben nur dann erforderlich, wenn eine spezielle Leitung getrennt werden soll. Dabei entsprechen die Bitnummern den Nummern der Interrupt-Eingänge.

Auch wenn ein Interrupt zur Zeit gesperrt ist, heißt das nicht, daß damit auch die Anforderung verloren gegangen ist. Die Anforderung für einen Interrupt wird im Anforderungsregister IR gespeichert und kommt dann zum Tragen, wenn durch erneutes Beschreiben des Operationswort 1 der Interrupt wieder freigegeben wird.

Bits 0 bis 7:

0: Der entsprechende Interrupt ist freigegeben.

1: Der entsprechende Interrupt ist gesperrt.

Wird beispielsweise der Wert 34_h (=0011 0100) in das Operationswort 1 geschrieben, sind die IR-Linien 2, 4 und 5 gesperrt.

Das Operationswort 2:

Mit Bit 1 des Initialisierungswort 4 wird entschieden, ob der 8249 auf ein End-Of-Interrupt-Befehl EOI von der CPU warten oder ob er seinen eigenen End-Of-Interrupt-Befehl AEOI erzeugen soll. Wenn dieses Bit im Initialisierungswort 4 gelöscht ist, wird das Operationswort 2 zum Senden des End-Of-Interrupt-Befehls an den 8259 benutzt. Andererseits kann das Operationswort 2 bei gesetztem Bit benutzt werden, um eine Prioritätsrotation bei einem AEOI durchzuführen. Des weiteren sind damit verschiedene Arten des EOI-Befehls ausführbar. Der ausgegebene EOI-Befehl kann spezifisch sein oder nicht spezifisch, d.h. man kann damit ein gewünschtes In-Service-Bit löschen oder automatisch das ISR-Bit rücksetzen, das die höchste Priorität besitzt.

Die Bits L0, L1 und L2:

Diese drei Bits des Operationsworts werden in Verbindung mit der Ausgabe eines spezifischen End-Of-Interrupt-Befehls EOI oder beim Aufbau einer neuen Prioritätsstruktur verwendet. Sie beinhalten die Zahlen von Null bis Sieben in binärkodierter Form und bezeichnen den IR-Pegel, dessen ISR-Bit gelöscht werden oder der die höchste Priorität erhalten soll (Tabelle 3-16).

L2	L1	L0	
0	0	0	IR-Pegel 0
0	0	1	IR-Pegel 1
0	1	0	IR-Pegel 2
0	1	1	IR-Pegel 3
1	0	0	IR-Pegel 4
1	0	1	IR-Pegel 5
1	1	0	IR-Pegel 6
1	1	1	IR-Pegel 7

Tabelle 3-16. Gesetzter Interrupt-Pegel

Die Bits EOI, SL und R:

Diese drei Bits bestimmen die Handhabung von EOI- oder AEOI-Befehlen. Nur drei der Kombinationen in Tabelle 3-17 benutzen die Bits L0 - L2. Es sind die Befehle, die das Bit SL auf Eins setzen. Sie sind zur Unterscheidung mit einem Stern (*) versehen. Bei den restlichen Ausführungen ist der Zustand der Bits L0 - L2 bedeutungslos.

R	SL	EOI	
0	0	0	Normaler EOI-Befehl
0	0	1	* Spezifischer EOI-Befehl
0	1	0	Rotation bei normalen EOI-Befehl
0	1	1	Einschalten der Rotation im AEOI-Modus
1	0	0	Ausschalten der Rotation im AEOI-Modus
1	0	1	* Rotation beim spezifischen EOI-Befehl
1	1	0	* Priorität setzen
1	1	1	Keine Änderung

Tabelle 3-17. Rotations- und EOI-Modi

Das Operationswort 3:

Das Operationswort 3 übt zwei Hauptfunktionen aus: Es kontrolliert erstens den Interrupt-Status und zweitens die Interrupt-Maskierung.

Der Interrupt-Status kann durch das Auslesen des In-Service-Registers ISR und des Interrupt-Anforderungsregisters IRR überprüft werden. Mit Hilfe des Testbefehls (Poll Command) kann manuell die höchste Priorität im In-Service-Register festgestellt werden.

Bit 0 - RIS: Dieses Bit gestattet in Verbindung mit dem folgenden Bit RR die Auswahl des Registers, auf das in einem nachfolgenden Lesebefehl durch die CPU zugegriffen wird. D.h. bevor die CPU entweder das IRR- oder das ISR-Register lesen kann, muß die Voreinstellung in beiden Bits erfolgen.

0: Der nachfolgende Lesebefehl greift auf den Inhalt des Interrupt-Anforderungsregisters IRR zu.

1: Der nachfolgende Lesebefehl greift auf den Inhalt des In-Service-Registers ISR zu.

Mit dem Inhalt beider Register kann man erfahren, welche Interrupts anhängig und welche gerade in Bearbeitung sind.

Bit 1 - RR: Read Register. Das Bit wird benutzt, um den Register-Lesebefehl auszuführen. Wenn das Bit gesetzt ist, bekommt erst das Bit 0 seine Gültigkeit und kann gesetzt werden.

0: Ein Register-Lesebefehl wird nicht ausgeführt.

1: Der folgende Lesebefehl der CPU wird entweder den Inhalt des IRR- oder des ISR-Registers lesen.

Bit 2 - P: Poll Command. Mit diesem Bit wird der Testbefehl an den 8259 gesandt. Der nächste Lesezugriff auf den 8259 bewirkt die Ausgabe eines Test-Bytes, dessen Inhalt angibt, ob ein gültiger Interrupt anliegt und welche Nummer die höchste Priorität hat. Vor jedem Testbefehl muß dieses Bit neu gesetzt werden.

0: Der 8259 führt keinen Testbefehl aus.

1: Der Testbefehl wird ausgeführt.

Bit 3 und 4:
Mit den beiden Bits wird die Unterscheidung zwischen den Operationswörtern 2 und 3 und dem Initialisierungswort 1 vorgenommen. Zur Auswahl des Operationsworts 3 müssen A_0=0, Bit 3=1 und Bit 4=0 sein.

Bit 5 - SMM: Special Mask Mode. Mit diesem Bit wird der Spezial-Masken-Modus ein- oder ausgeschaltet. Es entfaltet nur dann seine Wirkung, wenn zuvor das ESMM-Bit auf Eins gesetzt wurde.

0: Ausschalten des Spezial-Masken-Modus.

1: Einschalten des Spezial-Masken-Modus.

Bit 6 - ESMM: Enable Special Mask Mode. Das Bit gibt die Auswahl des Spezial-Masken-Modus mit dem SMM-Bit frei.

0: Verhindert die Wirkung des SMM-Bits.

1: Erlaubt dem SMM-Bit, den Spezial-Masken-Modus zu kontrollieren.

Bit 7: Es muß für eine korrekte Arbeitsweise des 8259 gelöscht sein.

3.4.4 Anwendungen

3.4.4.1 Die Adressierung des 8259

Zwei Faktoren sind für eine korrekte Adressierung zu berücksichtigen. Schreib- und Lesezugriffe sind auf den Baustein nur dann möglich, wenn die Leitung $\overline{CS}$ zuvor an Masse gelegt wurde. Das geschieht im allgemeinen mit Hilfe eines Adreß-dekoders, der den Adreßbus in geeigneter Weise interpretiert. Als zweites wird auf die verschiedenen Register durch Variation des Zustands an der A_0-Leitung zugegriffen. In diesem Zusammenhang muß darauf hingewiesen werden, daß sich die Funktion Chip-Select $\overline{CS}$ nur auf die Bus-Operationen auswirkt. Auf die Eingänge IR und auf die Wirksamkeit des $\overline{INTA}$-Signals hat der Zustand dieses Eingangs keinen Einfluß.

Das Bild 3-71 zeigt die Adressierung eines einzelnen 8259-Interruptcontrollers (Vgl. Bilder 3-52, 3-53, 3-55 und 3-56).

Als Selektorbaustein wird der programmierbare Dekoder 82C338 als 3-zu-8-Dekoder verwendet, der auf die speziellen Anforderungen des Systems programmiert werden kann. Es ist jedoch auch ohne weiteres möglich, einen üblichen TTL-Binär-zu-Dezimal-Dekoder zu verwenden. Im Bild 3-71 wird das $\overline{CS}$-Signal einem Ausgang des 82C338 entnommen und dem 8259 zugeführt. Der Baustein wurde in diesem Beispiel so programmiert, daß der Eingang G1 high-aktiv und die Eingänge G2 bis G5 low-aktiv sind. Die Adreßeingänge A, B und C sind mit den Adreßbusleitungen A2, A3 und A4 verbunden. Die Adreßleitung A0 wird direkt zum Eingang A_0 des 8259 geführt. Die Adreßleitung A1 ist in diesem Beispiel nicht verwendet, da sehr häufig in einem System weitere Peripheriebausteine angeschlossen sind, die zwei Adreßeingänge besitzen und daher die Adreßleitungen A0 und A1 benutzen.

Somit gibt es vier verschiedene Adressen, die den 8259 ansprechen und von denen allerdings zwei die gleiche Wirkung haben unabhängig davon, ob die Leitung A1 high oder low ist.

Da in Bild 3-71 der Ausgang $\overline{Y6}$ vom 82C338 mit dem $\overline{CS}$-Eingang verbunden ist, hat das zur Folge, daß die Leitungen A Low-Pegel, B und C High-Pegel führen müssen, um den Ausgang $\overline{Y6}$ an Masse zu setzen. Verbunden mit A0 erscheint der 8259 an den I/O-Adressen 18 und 19_h bzw. mit der gleichen Wirkung an 1A und $1B_h$ vorausgesetzt, die höheren Adreßleitungen führen Low-Pegel.

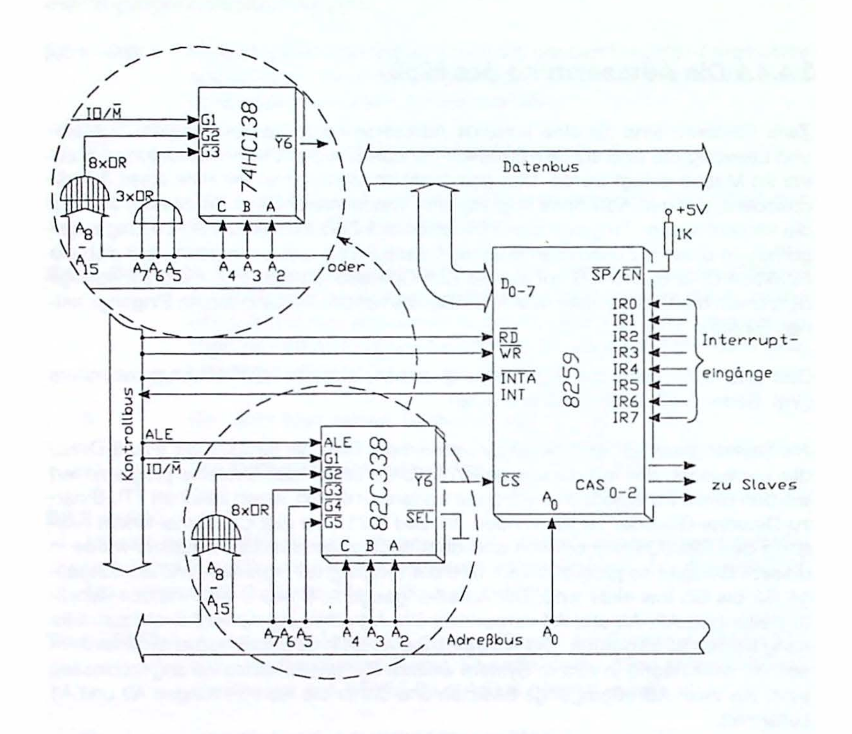


Bild 3-71. Die Adressierung des 8259

3.4.4.2 Erweiterung auf mehr als 64 Interrupts

In einigen Fällen mag es wünschenswert erscheinen, die Interrupts in einem System, das vielleicht für Kontrollzwecke ausgelegt ist, über die Maximalzahl von 64 hinaus zu erweitern. Die einfachste Art, das zu erreichen, ist die Benutzung des Testbefehls, um die Information über die Herkunft des Interrupts zu erhalten. Bild 3-72 zeigt in einem Beispiel, wie diese Erweiterung in einem 8088-System erreicht werden kann. Es werden zur Adressierung entweder ein 4-zu-16-Dekoder (74HC154) oder zwei 3-zu-8-Dekoderbausteine vom Typ 74HC138 oder 82C338 verwendet, um bis zu 16 Interruptcontroller zu adressieren. Keine weiteren Logikbausteine benötigt man bei der Verwendung von 82C338-Dekodern, da sie auf die spezielle Verwendung programmierbar sind. Im Beispiel nach Bild 3-72 sind alle Eingänge des ersten 82C338 low-aktiv, im zweiten ist lediglich der Eingang G2 high-aktiv. Das gestattet, beide Leitungen der Bausteine zusammenzufassen.

In dieser Art von Interrupt-Struktur wird weder das INT- noch das $\overline{\text{INTA}}$-Signal benötigt. Aus diesem Grund wird auch kein Interrupt am Prozessor ausgelöst, und es ist Aufgabe der Software, in geeigneten Zeitabständen mit einem Testbefehl bei jedem 8259 nach anhängenden Interrupts Ausschau zu halten. Ferner ist diese Art der Kommunikation die flexibelste, d.h. sie leistet auch in Zusammenarbeit mit den Mikrocontrollern gute Dienste. Die Leitung IO/$\overline{\text{M}}$ muß in solchen Fällen durch eine entsprechende ($\overline{\text{RD}}$ oder $\overline{\text{WR}}$) ersetzt werden.

Nachteile, die ein solches System mit sich bringt, sind der zusätzlich Zeitaufwand, den die Abfrage benötigt, und die Schwierigkeit, über diese Interrupt-Linien eine Realzeit-Anwendung zu realisieren. Letzteres kann jedoch dadurch gemildert werden, daß der Interrupt-Eingang der CPU jetzt frei ist und ein externer Timer-Baustein sowohl an einem 8259 als auch gleichzeitig an der CPU einen Interrupt erzeugen kann, so daß der Prozessor nicht ganz Zeit-los ist. Es hindert den Anwender ferner nichts, gesondert einen 8259 ohne Testbefehl zu verwenden, und ihn wie üblich mit der CPU über die Leitungen INT und $\overline{\text{INTA}}$ kommunizieren zu lassen.

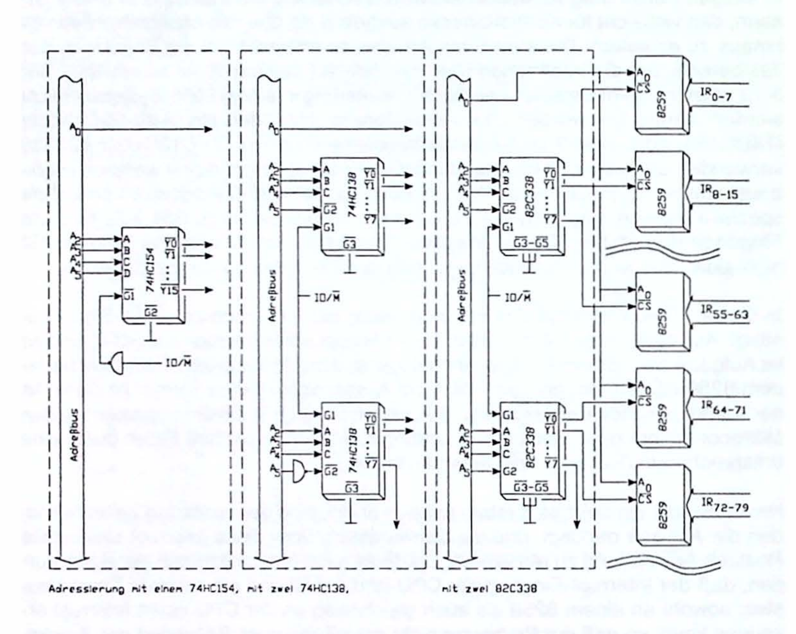


Bild 3-72. Die Erweiterung von mehr als 64 Interrupts

3.4.4.3 Die Interrupt-Versorgung unter MS-DOS

Der programmierbare Interruptcontroller 8259 ist im PC-System unter den I/O-Adressen 20h-21h zu finden, ein eventuell vorhandener zweiter Baustein (im AT) an den Adressen A0h-A2h. An den Interrupt-Eingängen sind die wichtigsten Peripheriebausteine angeschlossen (Tabelle 3-18). Der Timer-Baustein 8253 ist mit dem Eingang IR0 verbunden und löst 18,2 mal in der Sekunde den Timer-Interrupt aus, der zur Aktualisierung der internen Uhr dient. Die Tastatur wird durch einen eigenen Prozessor, den 8048 bzw. 8049 aus der Mikrocontroller-Familie MCS-48, überwacht, der eine Portleitung zu dem Eingang IR1 besitzt und dort einen registrierten Tastendruck anmeldet. Der dritte Eingang IR2 wird nicht benutzt. An den Eingängen IR3 und IR4 wird ein Interrupt erzeugt, der seinen Ursprung in der seriellen Datenübertragung hat. Die Festplatte erzeugt einen Interrupt an IR5 und das Diskettenlaufwerk an IR6. IR7 letztlich ist mit der Centronics-Schnittstelle verbunden, an der ein angeschlossener Drucker einen Interrupt auslösen kann.

Vektoradresse	Interrupt-Nr	IR-Eingang	Peripherie
20 - 23	8	0	Timer-Baustein 8253 (Echtzeituhr)
24 - 27	9	1	Mikrocontroller 8048 (Tastatur)
28 - 2B	A	2	frei
2C - 2F	B	3	COM2 (Serieller Port)
30 - 33	C	4	COM1 (Serieller Port)
34 - 37	D	5	Festplattenkontroller
38 - 3B	E	6	Floppy-Disk-Kontroller
3C - 3F	F	7	Drucker

Tabelle 3-18. Interrupt-auslösende Hardware im PC-System

3.4.4.4 Timer-erzeugte Interrupts

In einer Vielzahl von Controller-Anwendungen müssen zeitkritische Anwendungen durch das Programm erfüllt werden. Solche zeitabhängigen Aufgaben beinhalten Tastaturabfragen, Anzeigen, Monitoransteuerung, Druckerkontrolle und die verschiedensten Facetten von industriellen Aufgaben. Diese Beispiele sind ein Ausschnitt von vielen Entwicklungen, die eine Peripheriebedienung mit einer festen Zeitrate oder mit einer genauen Zeitverzögerung benötigen. Würde diese Aufgabe die CPU alleine ausführen, hätte das eine Verlangsamung des Datendurchsatzes und eine nicht mehr vertretbare Komplizierung der Software zur Folge. Diese Aufgabe erleichtert dem Prozessor der Interruptcontroller 8259 und der Timer-Baustein 8253.

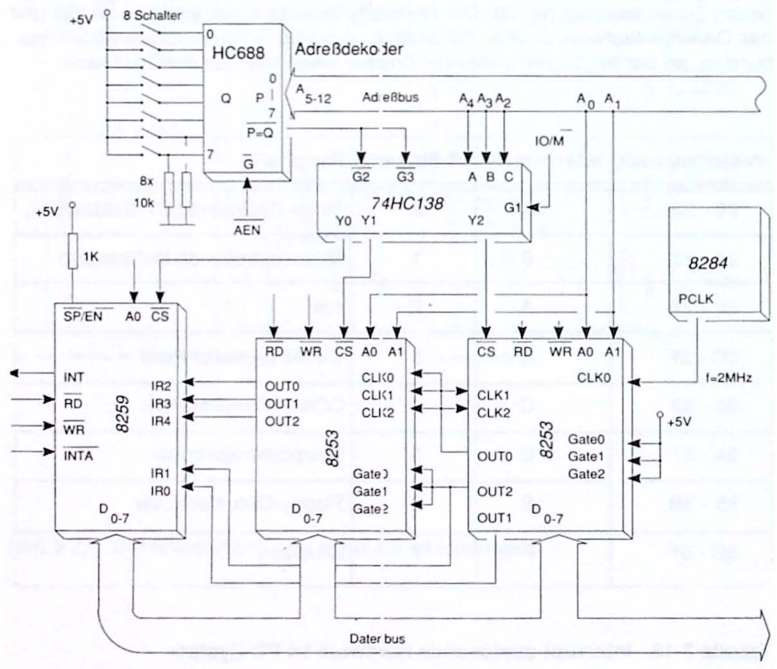


Bild 3-73. Durch den Timer 8253 erzeugte Interrupts

Das Anwendungsbeispiel nach Bild 3-73 verwendet den 8259 für die zeitgebenden Interrupts. Der externe Zeitablauf wird durch zwei Timer-Bausteine 8253#1 und 8253#2 geregelt. Ferner ist die notwendige I/O-Dekodierung durch den 74HC138 und einer Adreßlogik ebenfalls gezeigt.

Die 8253er werden in der Weise programmiert, daß sie in einer zeitlich konstanten Abfolge an den verschiedenen Eingängen des 8259 Interrupts auslösen, die der CPU die Basis zur Realzeit geben. Jeder 8253 verfügt intern über drei Zähler, die durch den Prozessor programmierbar sind. Das Ausgangssignal der Zähler ist eine Funktion des Takteingangssignals, dem programmierten Modus, dem voreingestellten 16-Bit-Zählwert und dem Pegel des Eingangs Gate.

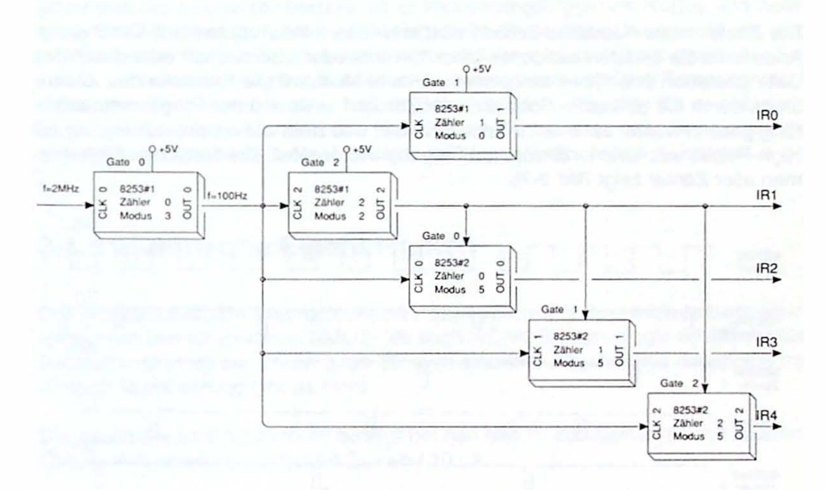


Bild 3-74. Die Zählersteuerung für zeitlich kontrollierte Interrupts

Bild 3-74 zeigt in überschaubarer Weise das Zusammenwirken der einzelnen Zähler. Das Beispiel geht davon aus, daß das System von einem 12-MHz-Quarz und dem Oszillatorbaustein 8284 betrieben wird. Somit steht ein PCLK-Signal von 2 MHz zur Verfügung, das dem Zähler 0 des 8253#1 zur Teilung durch 20000 zugeführt wird. Daraus wird ein Signal von 100 Hz mit einer Periodendauer von 10 ms, das die Taktbasis für die restlichen Zähler bildet. Mit dieser 10-ms-Zeitbasis ist es nun leicht, für die anderen Zähler entsprechende Teilungswerte zu errechnen. Zähler 2 des 8253#1 wird in Modus 2 als Ratengenerator benutzt. Er wird so programmiert, daß er alle zwei Sekunden - das entspricht 200 Taktimpulsen - einen Puls von 10 ms Breite ausgibt. Sein Ausgang erwirkt am Eingang IR1 des 8259

einen Interrupt. Alle Zähler des 8253#2 werden in Modus 5 als hardware-ausgelöster Takt betrieben, in dem der Eingang Gate die Zähleroperation startet. In diesem Beispiel kontrolliert der Ausgang von Zähler 2 des 8253#1 die Gates der Zähler 8253#2. Wenn an einen dieser Zähler ein Impuls von Baustein 8253#1 gelangt, gibt er selbst einen Impuls von 10 ms Länge aus, sobald sein Inhalt durch den Takt auf Null gezählt wurde.

Die programmierten Taktpulse sind für die Zähler des 8253#2 sind wie folgt: 50 Impulse (0,5 s) für Zähler 0, 100 Impulse (1 s) für Zähler 1 und 150 Impulse (1,5 s) für Zähler 2. Die Zählerausgänge erzeugen Interrupts an den Eingängen IR2 bis IR4 des 8259.

Der Zähler 1 des Bausteins 8253#1 wird in Modus 0 Interrupt bei Null-Durchgang. Anders als die anderen benutzten Modi, die entweder automatisch oder durch das Gate gesteuert ihre Arbeit aufnehmen, erlaubt Modus 0 die Kontrolle des Zählerstarts durch die Software. Sobald dieser Zähler 1 während der Programmausführung gesetzt wurde, zählt er 25 Takte (250 ms) und zieht dann seinen Ausgang auf High-Pegel, was einen Interrupt am Eingang IR0 auslöst. Die zeitlichen Signalformen aller Zähler zeigt Bild 3-75.

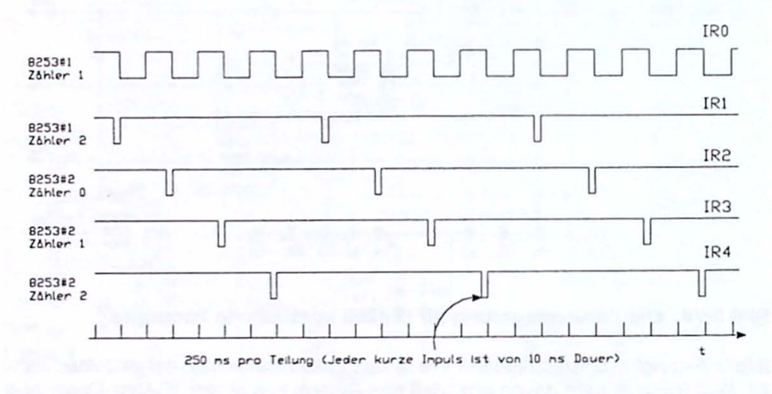


Bild 3-75. IR-Eingangssignale von den Timer-Bausteinen 8253

Es gibt prinzipiell zwei Methoden zur Erzeugung der Zeitbasis, wie sie in Zähler-orientierten Interrupt-Strukturen auftreten kann: abhängige Zeitgebung und unabhängige Zeitgebung. Die abhängige Zeitgebung benutzt eine einzige Zeitbasis und leitet daraus alle restlichen erforderlichen Zeitintervalle ab. Die unabhängige Zeitgebung hingegen kennt keine Wechselwirkung der zeitgebenden

Impulse. Für industrielle Kontrollanwendungen ist die abhängige Zeitgebung im allgemeinen die geeignetere, während die unabhängige mehr für den Einsatz in individuellen Geräten geeignet ist.

Im Beispiel nach Bild 3-74 sind beide Möglichkeiten verwirklicht. Die Interrupt-Erzeugung durch Zähler 1 im 8253#1 ist unabhängig von den restlichen Zählern, wohingegen die Interrupts 2, 3 und 4 durch Interrupt 1 beeinflußt sind. Jeder der Zähler des 8253#2 wird einen Interrupt nach einer bestimmten Zeitspanne erzeugen, sobald der Interrupt 1 erschienen ist.

Der 8259 seinerseits verarbeitet die ausgelösten Interrupts gemäß seiner Programmierung. In diesem Beispiel ist er im flankengetriggerten Modus und Automatic End-Of-Interrupt-Modus AEOI eingestellt. Unter dieser Voraussetzung wird an IR0 in jeder halben Sekunde, an IR 1 bis IR5 versetzt um jeweils eine halbe Sekunde, nach Ablauf von zwei Sekunden ein Interrupt erzeugt. Dem Prozessor steht somit eine Zeitbasis von einer halben Sekunde zur Verfügung, die er benutzen kann, um die interne Uhr und den Kalender zu aktualisieren oder beispielsweise eine Anzeigenleuchte mit der Frequenz von einem Hertz blinken zu lassen.

3.4.5 Elektrische Eigenschaften

Der programmierbare Interruptcontroller 8259 wird von zehn Halbleiterherstellern angeboten und ist sowohl in NMOS- als auch in CMOS-Technologie erhältlich. Alle Bausteine sind an der Zahlenfolge 8259 zu erkennen. Äquivalente Bausteine mit anderer Numerierung gibt es nicht.

Die maximale Stromaufnahme beträgt bei den NMOS-Bausteinen 85 mA, bei den CMOS-Versionen im statischen Zustand 10 µA.

Wichtig für die Zusammenarbeit mit den Prozessoren sind die minimalen Zeiten, die eingehalten werden müssen, damit auf den 8259 sicher zugegriffen werden kann. So müssen beim Schreiben ($\overline{WR}$) die Daten 240 ns vor dem Schreibimpuls, der eine Dauer von 290 ns haben muß, stabil anliegen. Der Leseimpuls ($\overline{RD}$ bzw. $\overline{INTA}$) kommt mit einer Dauer von 235 ns aus. Die Zeit, die zwischen zwei verschiedenen Befehlen verstreichen muß, beträgt 500 ns. Zwei aufeinanderfolgende $\overline{INTA}$-Sequenzen müssen durch 625 ns voneinander getrennt sein. Im flankengetriggerten Modus ist es wichtig, das IR-Signal für mindestens 100 ns an Masse zu halten.

Das sind die Daten für den langsamsten Baustein. Sie sind jedoch alle in etwas schnellerer Qualität erhältlich, was besonders für die CMOS-Versionen zutrifft. Hier liegen die Zugriffszeiten bei 160 ns, im Gegensatz zu 240 ns zu der NMOS-Version. Man mag nun sagen, das ist nicht gerade berauschend im Vergleich zu den Taktfrequenzen der modernen Prozessoren und RAMs. Diesem Nachteil wird von den Herstellern dadurch Rechnung getragen, daß der 80286 bei jedem ausgegegeben $\overline{\text{INTA}}$-Impuls einen Wait-State und zusätzlich drei weitere zwischen den beiden $\overline{\text{INTA}}$-Impulsen einlegt. Das ist in der Hardware des Systems und des Prozessors begründet und muß nicht durch die Software erfolgen.

Der 80386 reagiert in ähnlicher Weise mit dem Unterschied, daß zwischen den beiden $\overline{\text{INTA}}$-Signalen vier Wait-States liegen. Dadurch kann der 82C59 noch in Systemen mitarbeiten, die mit 24 MHz getaktet werden.

Kapitel 4

4. Bausteine der weiteren Prozessorperipherie

Zu den Bausteinen der weiteren Prozessorperipherie gehören solche, die nicht unmittelbar für eine Zusammenarbeit mit einem speziellen Prozessorensystem zugeschneidert sind, obwohl sie doch für das ein oder andere System unverzichtbar sind. So benötigt jede CPU I/O-Ports, mit deren Hilfe sie beispielsweise von der Tastatur Daten empfangen und allgemein mit der Umwelt in Verbindung treten kann. Das muß nicht ein ganz bestimmter Baustein sein, der ihr diese Kommunikation erst ermöglicht. In Frage kommen alle Port-Bausteine und man kann den für eine spezielle Schaltung am geeignetsten wählen. Aber benötigt wird in jedem Fall einer. Häufig sind die wichtigsten Peripheriefunktionen Dank des fortschreitenden Integrationsgrads bereits neben der CPU auf demselben Chip vorhanden und reduzieren die Größe einer Schaltung beträchtlich. Sie können sogar den Programmspeicher und einen kleinen Teil des Datenspeichers beinhalten, so daß außer einer Stromversorgung und einem Interface keine weiteren Teile benötigt werden. Das sind die Mikrocontroller. Daneben gibt es Zwischenstufen wie der 80188, der neben einer 8088-CPU bereits die wichtigsten Bausteine der engeren Peripherie besitzt. Auf zusätzliche I/O-Ports ist auch er angewiesen.

Die weiteren in diesem Kapitel vorgestellten Bausteine sind für das Funktionieren eines Systems nicht so immens wichtig wie ein I/O-Port-Baustein oder die Bausteine der engeren Peripherie, sie sind jedoch in der Lage, der CPU ständig wiederkehrende oder zeitintensive Aufgaben abzunehmen. Das entlastet nicht nur die CPU, sondern auch den Programmierer. Die entscheidendsten Vorteile sind jedoch die Erhöhung des Datendurchsatzes, der Geschwindigkeit, der Flexibilität und somit der Leistungsfähigkeit eines Systems. Man stelle sich vor, die CPU wollte ohne die Verwendung eines Videokontrollers die Steuersignale für einen Monitor ausgeben. Sie wäre damit so beschäftigt, daß sie keine weiteren Aufgaben mehr wahrnehmen könnte.

Die CPU steht als Manager über den Peripheriebausteinen und koordiniert ihre Arbeit. Diese sind also kein Luxus, sondern eine Notwendigkeit für ein modernes leistungsfähiges System.

4.1 Der programmierbare Intervall-Timer-Baustein 8253

4.1.1 Übersicht

Der programmierbare Zähler 8253 verfügt über einen 8-Bit-Datenbus D_0 bis D_7, zwei Adreßeingänge A_0 und A_1, Steuereingänge $\overline{WR}$, $\overline{RD}$ und $\overline{CS}$ sowie über drei unabhängige 16-Bit-Abwärtszähler, die über die Anschlüsse CLK, GATE und OUT mit der Umwelt in Kontakt treten. Die Zähler können maximal über die Eingänge CLK mit einer Taktfrequenz von 2,6 MHz betrieben werden. Mit den Steuer- und den Datenbusleitungen ist er zu allen Prozessoren der 80XX-Reihe kompatibel und wird wie ein gewöhnlicher I/O-Baustein von der CPU angesprochen (Bild 4-1).

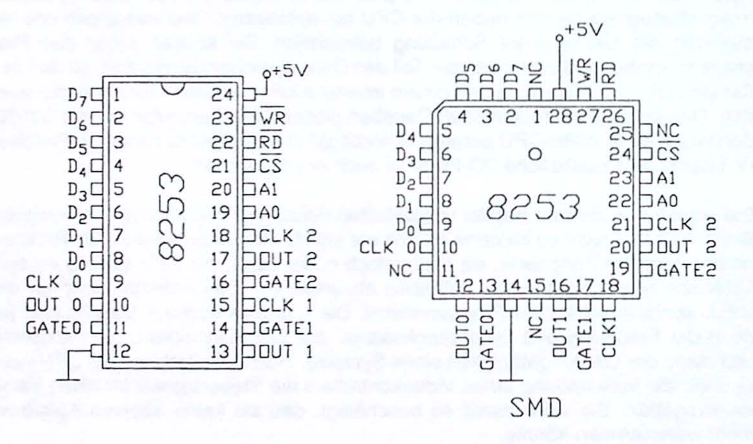


Bild 4-1. Pin-Belegung des 8253

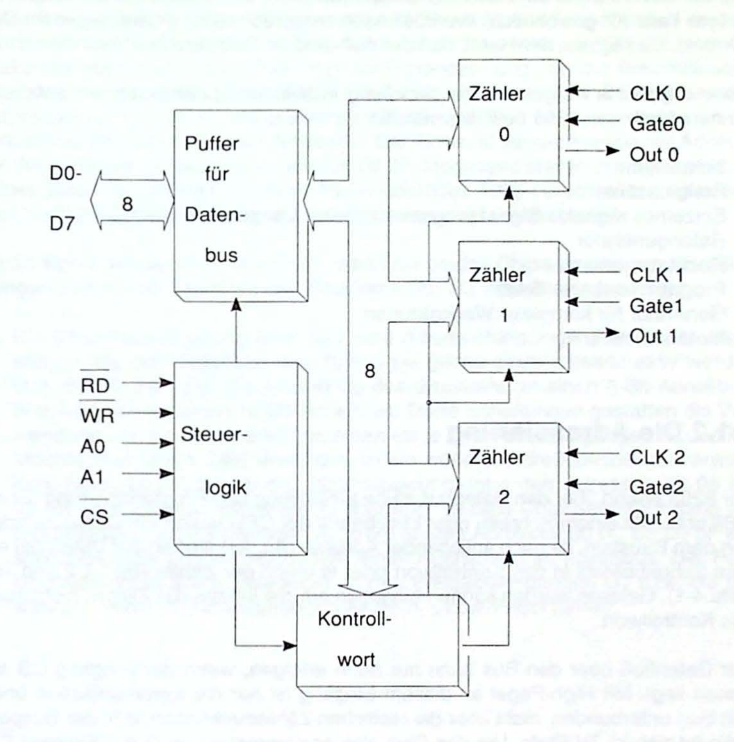


Bild 4-2. Blockschaltbild des 8253

Von seiner Funktion her ist er ein allgemeines und vielseitiges zeitbeeinflussendes Element, das wegen seiner Programmierbarkeit zu den unterschiedlichsten Aufgaben verwendet werden kann. Am häufigsten wird er wohl zur Lösung eines der gewöhnlichsten Probleme in einem Mikrocomputersystem verwendet werden: der Erzeugung von exakten, software-kontrollierten Zeitschleifen. Er entbindet die CPU von der Ausführung der benötigten Wartezeiten und setzt sie währenddessen für andere Aufgaben frei. Als Programmierer wird man diesen Vorteil rasch zu schätzen wissen und wird in diesem Falle einen Zähler des 8253 in der gewünschten Weise konfigurieren, ihn mit dem Startwert beschreiben und ihn abwärts mit der am Eingang CLK anliegenden Frequenz zählen lassen. Beim Zählerstand von Null

wird der 8253 bei der CPU einen Interrupt auslösen. Die Taktquelle entstammt in diesem Falle für gewöhnlich, eventuell nach entsprechender Vorteilung, dem Systemtakt. Es liegt auf der Hand, daß der Aufwand an Software hierbei minimal ist.

Weitere typische Aufgaben, wie sie häufig in Mikrocomputersystemen anfallen, können durch den 8253 bewältigt werden:

- Echtzeituhr
- Ereigniszähler
- Einzelnes digitales Signal programmierbarer Länge (One Shot)
- Ratengenerator
- Rechteckgenerator
- Programmierbarer Teiler
- Generator für komplexe Wellenformen
- Motorcontroller

4.1.2 Die Adressierung

Der 8253 erfährt über den Datenbus seine Einstellung und Programmierung. Er ist 8 Bit breit. Mit einem Schreib- oder Lesebefehl der CPU laufen die Daten zu oder von dem Baustein. Je nach anliegender Adresse (A_0, A_1) fließen die Daten bei einem Schreibbefehl in das Kontrollwort oder in einen der Zähler (Bild 4-2 und Tabelle 4-1). Gelesen werden können hingegen nur die Inhalte der Zähler, nicht aber das Kontrollwort.

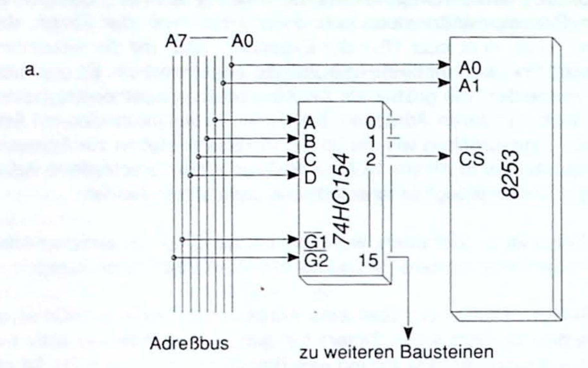
Der Datenfluß über den Bus kann nur dann erfolgen, wenn der Eingang $\overline{CS}$ an Masse liegt. Mit High-Pegel an diesem Eingang ist nur die Kommunikation über den Bus unterbunden, nicht aber die restlichen Zählerfunktionen, d.h. der Busport befindet sich im Tri-State. Um den Chip also anzusprechen, muß der Eingang $\overline{CS}$ auf eine der folgenden Weisen mit dem Adreßbus verbunden sein:

1. A_0, A_1 sind mit den korrespondierenden Adreßleitungen A_0 und A_1 verbunden. Chip-Select $\overline{CS}$ ist über einen Inverter an einer der folgenden Adreßleitungen A_3 bis A_{15} angeschlossen. Dieses Verfahren kann angewendet werden, wenn zahlenmäßig wenige, d.h. nicht mehr als vierzehn I/O-Bausteine dieser Art in einem System zum Einsatz kommen. Außerdem erfordert diese Hardware-Anordnung eine verantwortungsvolle Programmierung, da die Chips über bestimmte Adressen gleichzeitig ansprechbar sind.

In folgendem Beispiel ist die Chip-Select-Leitung eines Timers 8253 mit A_2 und die Chip-Select-Leitung eines I/O-Expanders 8255 mit A_3 des Adreßbusses verbunden. Mit der Bitkombination xxxx xxxx xxxx 01xx wird der Timer, mit der Bitkombination xxxx xxxx xxxx 10xx der Expander und mit der Bitkombination xxxx xxxx xxxx 11xx werden beide gleichzeitig angesprochen. Es sind also alle Adressen zu vermeiden, die größer als XX XB sind. Des weiteren erscheinen die Bausteine an vielen anderen Adressen. Der Timer ist beispielsweise mit Adresse 36 A4 genauso angesprochen wie mit 08 27. Insgesamt stehen zur Adressierung eines jeden Bausteines in einem 16-Bit-Adreßbus 4096 verschiedene Adressen zur Verfügung, die sich jedoch in ihrer Wirkung nicht unterscheiden.

In ähnlicher Weise ist zu verfahren, wenn die beiden Chips an anderen Adreßleitungen liegen oder wenn weitere Bausteine den I/O-Adreßbereich belegen.

2. Die Chip-Select-Leitung wird über eine Adressierungslogik (74HC154) angesteuert, die den Datenbus des Timers bei genau einer Adresse aktiv werden läßt. Bild 4-3 a zeigt die Anbindung des Bausteins an einen 8-Bit-Adreßbus, Bild 4-3 b die an einen 16-Bit-Adreßbus. Diese Schaltungen gestatten die Verwendung von 64 Peripheriebausteinen mit je zwei Adreßeingängen A_0 und A_1. Möchte man diese Zahl erweitern, ist ein weiterer Adreßdekoder zu verwenden. Nach Bild 4-3 b ist der 8253 eindeutig unter den Adressen 00 08 bis 00 0B zu erreichen; ein weiterer Baustein, dessen Chip-Select-Eingang mit Dekoderausgang 3 verbunden ist, wäre an den Adressen 00 0C bis 00 0F ansprechbar. Die Dioden verhindern einen Kurzschluß im Adreßbus. Der Pull-Up-Widerstand soll Massepegel an den Freigabeeingängen G1 bzw. G2 gewähren, wenn die Adreßleitungen A_6 bis A_{15} Low-Pegel führen.



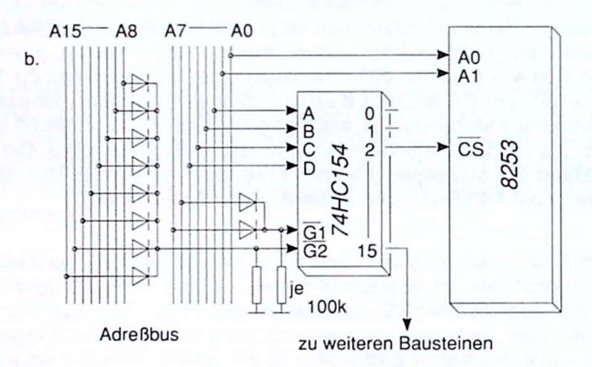


Bild 4-3. Anschluß des 8253 an einen 8-Bit- bzw. 16-Bit-Adreßbus

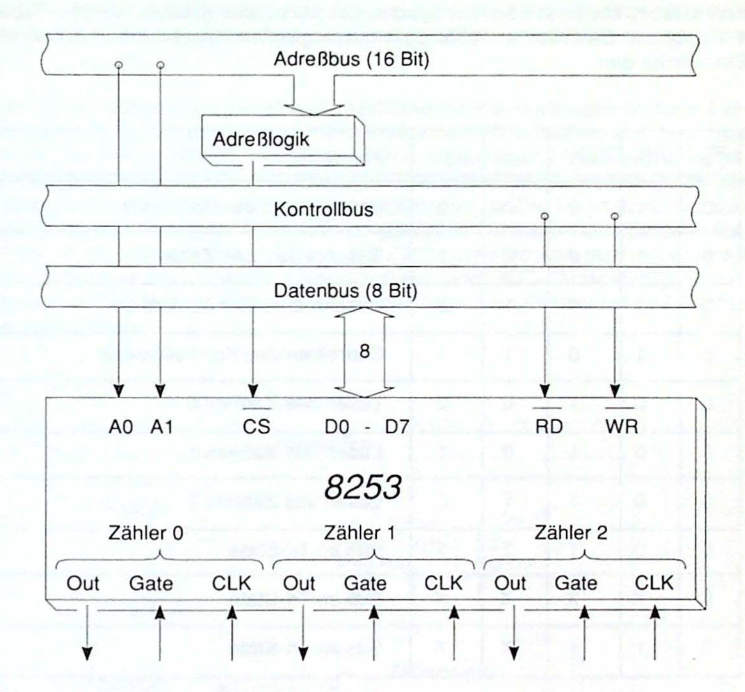


Bild 4-4. 8253-System-Interface

Neben dem Busfreigabe-Pin $\overline{CS}$ existiert noch der Schreibeingang $\overline{WR}$ bzw. der Leseeingang $\overline{RD}$. Diese Steuereingänge werden einfach mit den gleichnamigen Steuerleitungen der CPU bzw. des Buscontrollers verbunden (Bild 4-4). Mit einem Schreibbefehl führt die $\overline{WR}$-Leitung, mit einem Lesebefehl die $\overline{RD}$-Leitung Massepegel und ist dadurch aktiviert.

Wohin bzw. woher die Daten kommen, wird mit den logischen Pegeln an den Eingängen A0 und A1 geregelt. Liegen beide an Null, ist Zähler 0 angesprochen, mit A0=1 und A1=0 Zähler 1, mit A0=0 und A1=1 Zähler 2 und mit A0=1 und A1=1 das Kontrollwort. Letzteres kann nur beschrieben, nicht aber gelesen werden. Tabelle 4-1 zeigt den Datenfluß in Abhängigkeit der logischen Pegel auf den Adreß- und Steuerleitungen.

$\overline{CS}$	$\overline{RD}$	$\overline{WR}$	A1	A0	Funktion
0	1	0	0	0	Beschreiben des Zählers 0
0	1	0	0	1	Beschreiben des Zählers 1
0	1	0	1	0	Beschreiben des Zählers 2
0	1	0	1	1	Schreiben des Kontrollwortes
0	0	1	0	0	Lesen des Zählers 0
0	0	1	0	1	Lesen des Zählers 1
0	0	1	1	0	Lesen des Zählers 2
0	0	1	1	1	Bus im Tri-State
0	X	X	X	X	Bus im Tri-State
0	1	1	X	X	Bus im Tri-State

Tabelle 4-1. Datenfluß in Abhängigkeit von Adreß- und Steuerleitungen im 8253

4.1.3 Die Zähler

Der 8253 besitzt drei Zähler, die mit 0, 1 und 2 numeriert werden. Bezüglich ihrer Arbeitsweise sind sie identisch, so daß im folgenden nur ein Zähler beschrieben wird.

Jeder Zähler besteht aus einem voreinstellbaren 16-Bit-Abwärtszähler, der mit einem externen Signal am Eingang CLK getaktet wird. Die Taktfrequenz kann von Null bis 2,6 MHz variieren, so daß der Zähler als externer Ereigniszähler eingesetzt werden kann. Man achte jedoch auf hinreichende Flankensteilheit, da der Eingang CLK nicht level-, sondern flankengetriggert ist. Die Minimalzeit für Low-Pegel ($U<0{,}8$ V) ist 150 ns, für High-Pegel ($U>2{,}2$ V) 230 ns. Mit dem Eingang Gate kann je nach eingestelltem Modus der Zähler unterbrochen oder neugeladen werden. Darüber hinaus ist der Zähler als Binär- oder BCD-Zähler konfigurierbar. Lädt man ihn mit Null, so kann er die höchste Zählrate ausführen; das ist 2^{16} binär oder 10^4 dezimal.

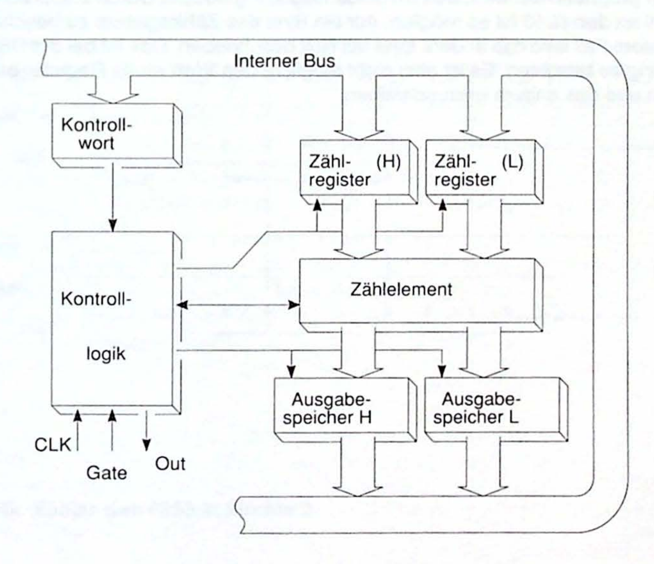


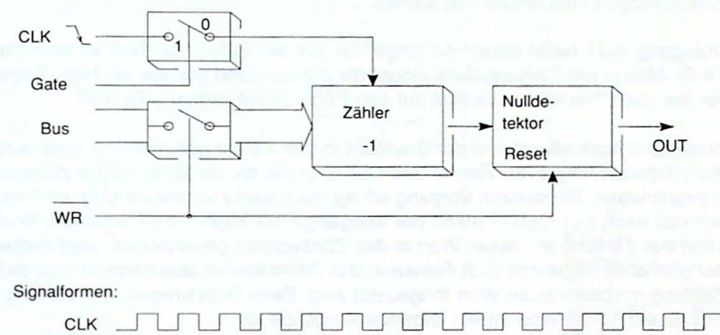
Bild 4-5. Blockdiagramm eines Zählers 8253

Durch Beschreiben des Kontrollwortes kann der Zähler in einen von sechs verschiedenen Modi eingestellt werden. Im Kontrollwort ist in den Bits 6 und 7 die Zähleradresse enthalten, in den restlichen Bits die Modusinformation, die an den Zähler weitergereicht und dort aufbewahrt wird (Bild 4-5). Somit braucht ein Zähler nur einmal eingestellt zu werden. Eine Umprogrammierung ist jederzeit möglich, ebenso ein Lesen des Zählerinhaltes, ohne Beeinflussung des Zählerstandes und ohne daß der Zähler dazu gestoppt werden müßte. Da der interne Bus 8 Bit umfaßt, sind zwei Zählregister und zwei Ausgabespeicher vorhanden. In der Regel folgt der Ausgabespeicher dem Inhalt des Zählelements, aber wenn ein spezieller Speicherbefehl an den 8253 gelangt, wird die Kopierung des Zählelementes gestoppt, bis der gespeicherte Wert von der CPU gelesen ist. Ohne diesen speziellen Befehl wird der aktuelle Zählerstand gelesen. Durch entsprechende Einstellung des Kontrollwortes kann auch nur einer von beiden Ausgabespeicher gelesen werden.

Wird der Zähler beschrieben, so muß der Schreibvorgang wegen des 8-Bit-Busses zweimal erfolgen. Beide Bytes gelangen nacheinander in die beiden Zählregister, von wo aus sie parallel als 16-Bit-Wort das Zählelement voreinstellen. Wenn der Zähler programmiert wird, werden beide Register gelöscht. Durch entsprechenden Befehl an den 8253 ist es möglich, nur ein Byte des Zählregisters zu beschreiben. In diesem Fall wird das andere Byte mit Null beschrieben. Das ist bei der Programmierung zu beachten. Es ist also nicht möglich, den Wert eines Register beizubehalten und das andere umzuschreiben.

4.1.4 Die Betriebsarten der Zähler

4.1.4.1 Modus 0: Interrupt bei Zählerendstand



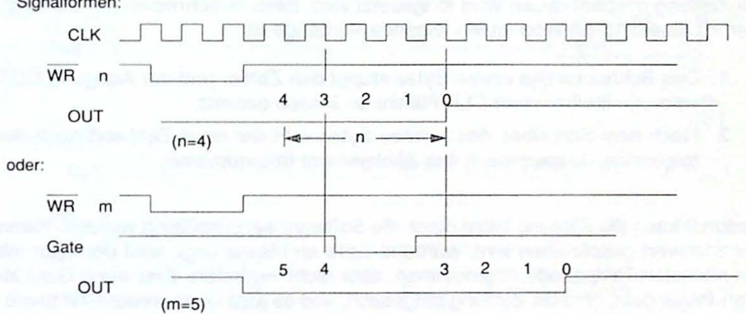


Bild 4-6. Zähler des 8253 in Modus 0

Modus 0 wird üblicherweise zum Zählen von externen Ereignissen benutzt. Das Erreichen des Zählwertes Null wird durch eine positive Flanke am Ausgang OUT angezeigt. Da dieser Pegelwechsel häufig zum Auslösen eines Interrupts verwendet wird, hat der Modus diesen Namen erhalten.

Der Ausgang OUT kann durch zwei Maßnahmen an Masse gesetzt werden:

1. Beschreiben des Kontrollwortes mit Modus 0,
2. Schreiben eines neuen Zählwertes.

Der Ausgang OUT bleibt darauf so lange low, bis der Zähler die Null erreicht hat (Bild 4-6). Masse am Eingang Gate stoppt die Zählung und gibt sie mit High-Pegel wieder frei. Gate hat keinen Einfluß auf den Pegel des Ausgangs OUT.

Nachdem das Kontrollwort und der Startwert in den Zähler geschrieben sind, wird mit der nächsten negativen Flanke des CLK-Signals dieser Wert in das Zählelement geschrieben. Bei diesem Vorgang erfolgt noch keine Verminderung um Eins, so daß erst nach n+1 CLK-Flanken der Ausgang OUT High-Pegel annimmt. Wird während der Zählung ein neuer Wert in das Zählregister geschrieben, wird dieser mit der nächsten negativen CLK-Flanke in das Zählelement übernommen, so daß die Zählung mit dem neuen Wert fortgesetzt wird. Beim Beschreiben der Zählregister mit einem 16-Bit-Wort laufen folgende Vorgänge ab:

1. Das Schreiben des ersten Bytes stoppt den Zähler und der Ausgang OUT wird unmittelbar ohne CLK-Flanke an Masse gesetzt.
2. Nach dem Schreiben des zweiten Bytes wird der neue Zählwert nach der folgenden Taktperiode in das Zählelement übernommen.

Dadurch kann die Zählung leicht durch die Software synchronisiert werden. Wenn der Startwert geschrieben wird, während Gate an Masse liegt, wird der Wert mit der nächsten Taktperiode übernommen, aber nicht verändert. Erst wenn Gate an High-Pegel geht, wird die Zählung fortgesetzt, und es wird keine weitere Taktperiode zum Laden benötigt, da dies ja bereits erfolgt ist. OUT geht n Taktperioden später an Plus.

Es sei ausdrücklich darauf hingewiesen, daß nach einem Zählerunterlauf das Zählelement nicht automatisch mit dem Inhalt der Zählregister nachgeladen wird und der Ausgang OUT High-Pegel beibehält. Ein Neustart wird nur durch das Beschreiben des Kontrollwortes mit Modus 0 oder mit dem Laden der Zählregister ausgelöst.

4.1.4.2 Modus 1: Hardware-retriggerbarer One-Shot

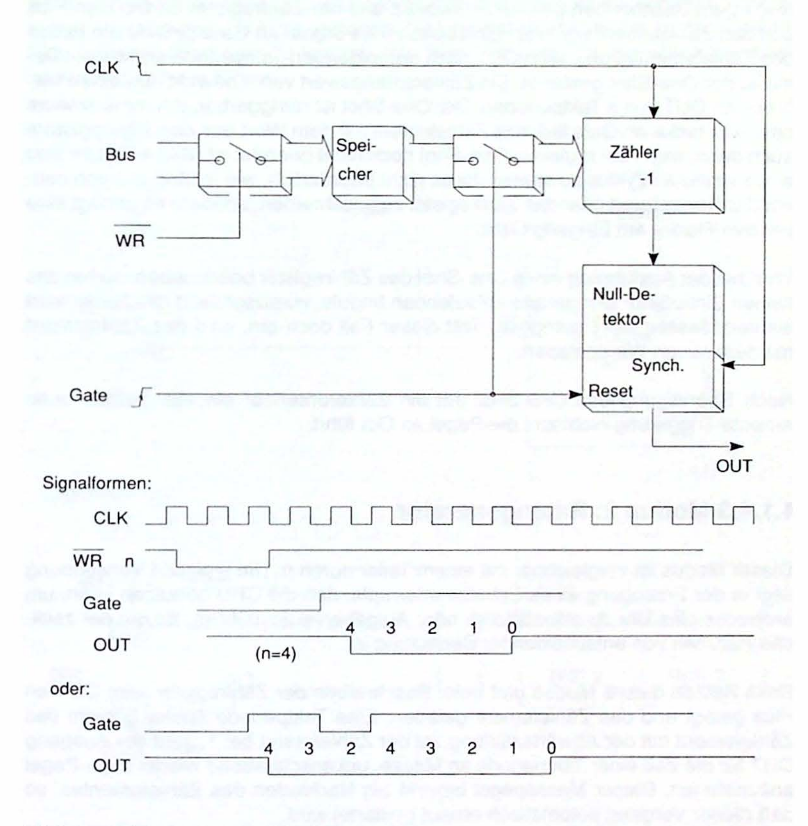


Bild 4-7. Zähler des 8253 in Modus 1

Unter einem One-Shot versteht man einen einmaligen Impuls, der durch ein externes Ereignis ausgelöst wird. In diesem Falle ist eine positive Flanke am Eingang Gate der Auslöser. Nach einer Taktperiode geht OUT an Masse, der Zähler zählt bis Null und OUT geht wieder an Plus. Dieser Vorgang wiederholt sich nicht automatisch, sondern bedarf eines Gate-Signals als Trigger.

Nach dem Beschreiben des Kontrollwortes und der Zählregister ist OUT an Plus und das Zählelement startklar. Ein auslösendes Signal an Gate bewirkt ein Laden des Zählelementes und setzt OUT nach der folgenden Taktperiode an Masse. Damit ist der One-Shot gestartet. Ein Zähleranfangswert von n bewirkt nun eine Pulsbreite an OUT von n Taktperioden. Der One-Shot ist retriggerbar, d.h. eine erneute positive Flanke an Gate lädt das Zählelement mit dem Wert aus den Zählregistern auch dann, wenn der laufende One-Shot noch nicht beendet ist (Bild 4-7). Um also einen erneuten Zyklus zu starten, ist es nicht erforderlich, wie in Modus 0 von neuem das Kontrollwort oder das Zählregister zu beschreiben, sondern es genügt eine positive Flanke am Eingang Gate.

Wird bei der Ausführung eines One-Shot das Zählregister beschrieben, so hat das keinen Einfluß auf den gerade ablaufenden Impuls, vorausgesetzt der Zähler wird währenddessen nicht getriggert. Tritt dieser Fall doch ein, wird das Zählelement mit dem neuen Wert geladen.

Nach Beendigung des One-Shot tritt ein Zählerunterlauf ein, der jedoch ohne erneute Triggerung nicht zu Low-Pegel an Out führt.

4.1.4.3 Modus 2: Ratengenerator

Dieser Modus ist vergleichbar mit einem Teiler durch n. Die typische Verwendung liegt in der Erzeugung eines Echtzeit-Interrupts, den die CPU benutzen kann, um entweder eine Uhr zu aktualisieren oder Aufgaben auszuführen, für die der zeitliche Rahmen von entscheidender Bedeutung ist.

Beim Setzen dieses Modus und beim Beschreiben der Zählregister wird OUT an Plus gelegt und das Zählelement geladen. Eine Taktperiode später beginnt das Zählelement mit der Abwärtszählung. Ist der Zählerstand bei 1, geht der Ausgang OUT für die Zeit einer Taktperiode an Masse, um anschließend wieder High-Pegel anzunehmen. Dieser Massepegel bewirkt ein Nachladen des Zählelementes, so daß dieser Vorgang automatisch erneut gestartet wird.

Diese Wiederholung kann nur durch Änderung des Modus oder durch Masse an Gate gestoppt werden. Wenn Gate während Low-Pegel an OUT an Masse geht, wird dieser Ausgang unmittelbar auf High-Pegel gesetzt. Eine positive Flanke an

Gate bewirkt ein Nachladen des Zählelementes und kann so zur Synchronisierung von Zähler und peripheren Bausteinen verwendet werden. Durch geschicktes zeitliches Beschreiben der Zählregister ist eine Synchronisierung durch die Software möglich.

Tritt während einer Zählsequenz eine Neubeschreibung der Zählregister ein, wird der neue Wert erst in der nächsten Zählperiode oder nach einem erfolgten Triggersignal übernommen. Auf die laufende Zählung hat die Änderung keinen Einfluß. Eine Voreinstellung des Zählers mit dem Wert 1 ist unsinnig und würde zu keiner Aktion des Zählers führen.

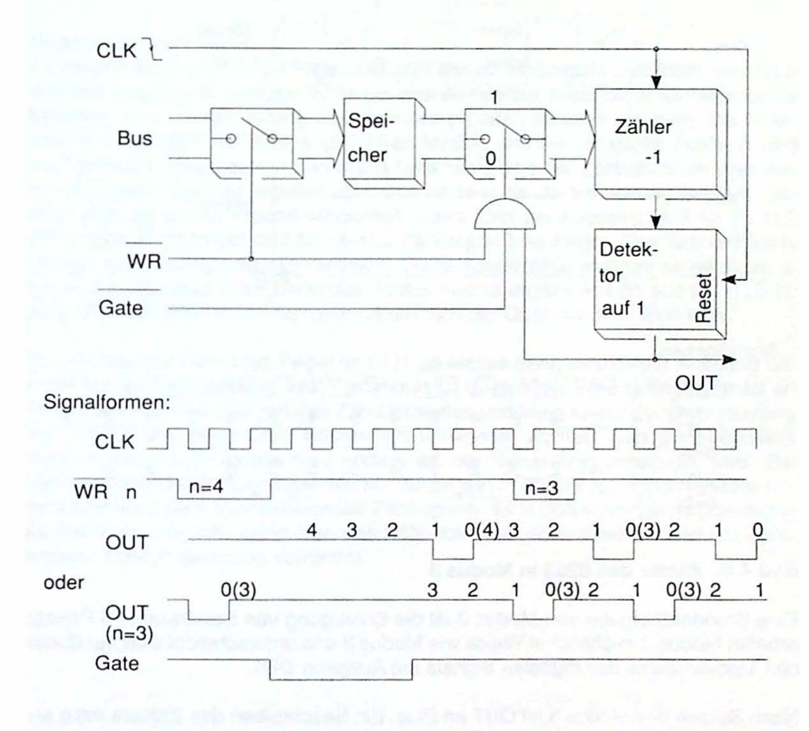


Bild 4-8. Zähler des 8253 in Modus 2

4.1.4.4 Modus 3: Rechteckgenerator

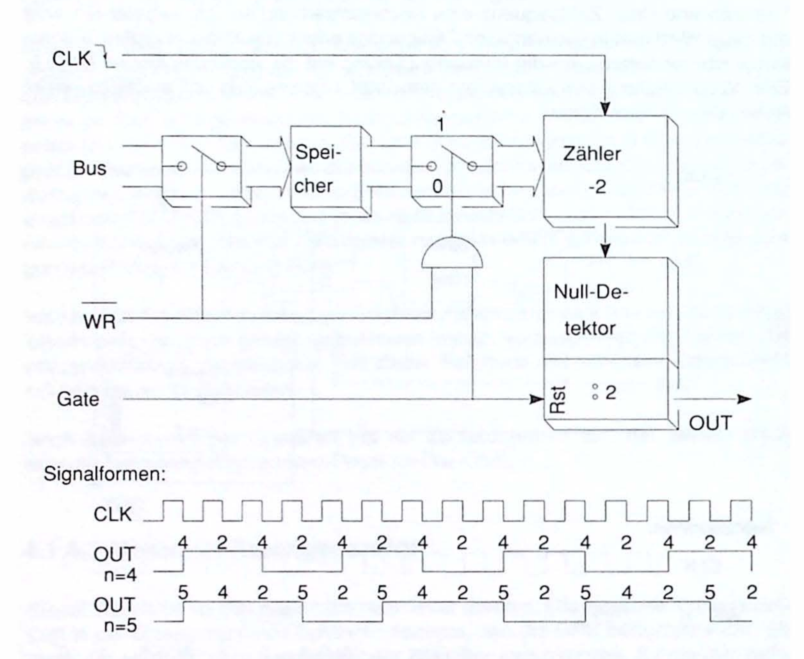


Bild 4-9. Zähler des 8253 in Modus 3

Eine Standardaufgabe von Modus 3 ist die Erzeugung von Baudraten. Im Prinzip arbeitet Modus 3 in ähnlicher Weise wie Modus 2 und unterscheidet sich nur durch das Tastverhältnis des digitalen Signals am Ausgang OUT.

Nach Setzen des Modus 3 ist OUT an Plus. Ein Beschreiben des Zählers mit n erzeugt ein Rechtecksignal mit der Periodendauer von n Taktzyklen. Nach n Taktzyklen geht OUT an Masse, anschließend wieder an Plus. Diese Abfolge wird kontinuierlich wiederholt.

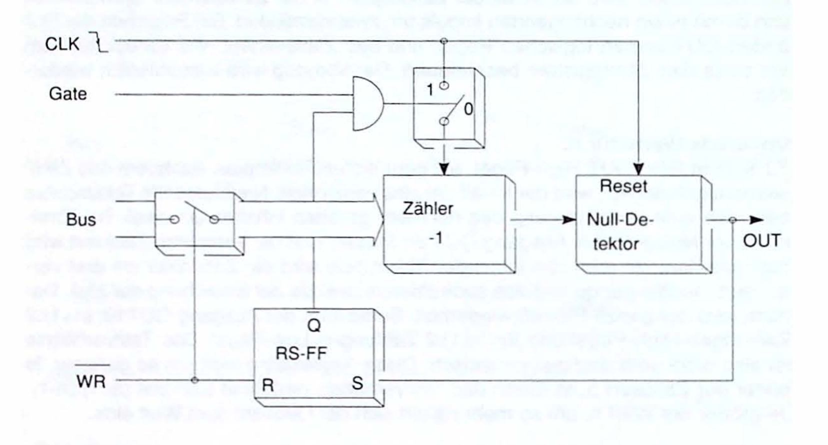
Um auch ungerade Werte für n zuzulassen, erfolgt die Durchführung der Zählung mittels Teilung durch zwei:

Gerade Werte für n:
Zu Beginn führt OUT High-Pegel. Mit dem ersten Taktimpuls nach dem Beschreiben des Zählers wird der Inhalt der Zählregister in das Zählelement übernommen und durch einen nachfolgenden Impuls um zwei vermindert. Bei Erreichen der Null ändert OUT seinen logischen Pegel, und das Zählelement wird erneut mit dem Wert aus den Zählregistern beschrieben. Der Vorgang wird kontinuierlich wiederholt.

Ungerade Werte für n:
Zu Beginn führt OUT High-Pegel. Mit dem ersten Taktimpuls, nachdem das Zählelement geladen ist, wird der Inhalt um eins vermindert. Nachfolgende Taktimpulse bewirken eine Verminderung des nunmehr geraden Inhaltes um zwei. Bei Erreichen der Null geht der Ausgang OUT an Masse, und der komplette Zählwert wird nachgeladen. Mit dem nun folgenden Taktimpuls wird der Zählinhalt um **drei** vermindert, nachfolgende Impulse subtrahieren zwei bis zur Erreichung der Null. Danach wird der ganze Prozeß wiederholt. Somit führt der Ausgang OUT für (n+1):2 Zählungen High-Pegel und für (n-1):2 Zählungen Low-Pegel. Das Tastverhältnis ist also nicht vollständig symmetrisch. Diese Asymmetrie wird um so geringer, je höher der Zählwert n ist. Denn das Tastverhältnis errechnet sich aus (n+1):(n-1). Je größer der Wert n, um so mehr nähert sich der Quotient dem Wert eins.

Wenn Gate während Low-Pegel an OUT an Masse geht, wird dieser Ausgang unmittelbar auf High gesetzt, kein Taktimpuls ist dazu nötig. Eine positive Flanke an Gate bewirkt ein Nachladen des Zählelementes und kann so zur Synchronisierung von Zähler und peripheren Bausteinen verwendet werden. Das gilt besonders dann, wenn als Taktquelle eine andere als der Systemtakt verwendet wird. Zur Synchronisierung genügt dann ein kurzer negativer Impuls am Eingang Gate unmittelbar nach dem Beschreiben der Zählregister. Eine Software-Synchronisierung ist ebenfalls möglich, wenn man das Schreiben des Kontrollwortes und der Zählregister zeitlich geschickt vornimmt.

4.1.4.5 Modus 4: Software-gesteuerter Taktimpuls



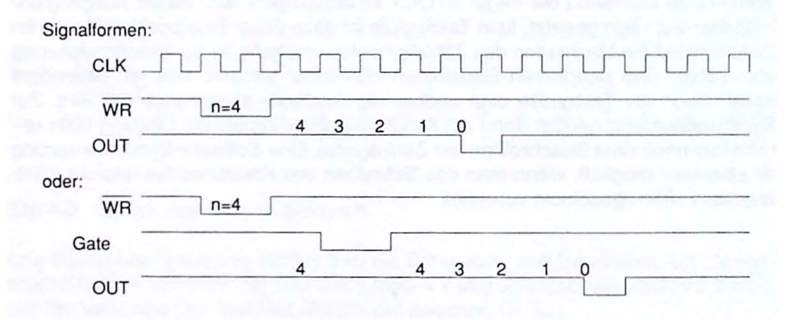


Bild 4-10. Zähler des 8253 in Modus 4

Vieles erinnert an das Verhalten des Zählers in Modus 0. In der Tat unterscheiden sich beide Modi nur in einem einzigen Punkt, das ist das Verhalten des Ausgangs OUT. Während in Modus 0 der Pegel des Ausgangs bei Nullstand des Zählers von Masse nach Plus wechselt, geht OUT in Modus 4 bei Nullstand an Masse und kehrt nach einem CLK-Zyklus nach Plus zurück.

Beim Setzen von Modus 4 geht der Ausgang Out an Plus. Der Zähler startet erst, wenn er mit einem Wert beschrieben wurde. Das Zählelement wird mit dem folgenden Taktsignal beschrieben, so daß erst der zweite Taktimpuls zur Verminderung des Inhaltes führt. Es vergehen somit (n+1) Taktzyklen nach dem Beschreiben des Zählinhaltes.

Wird während der Zählung ein neuer Wert geschrieben, verhält sich der Zähler wie in Modus 0.

High-Pegel an Gate gibt die Zählung frei, Low-Pegel stoppt die Zählung, ohne den logischen Pegel an OUT zu beeinflussen.

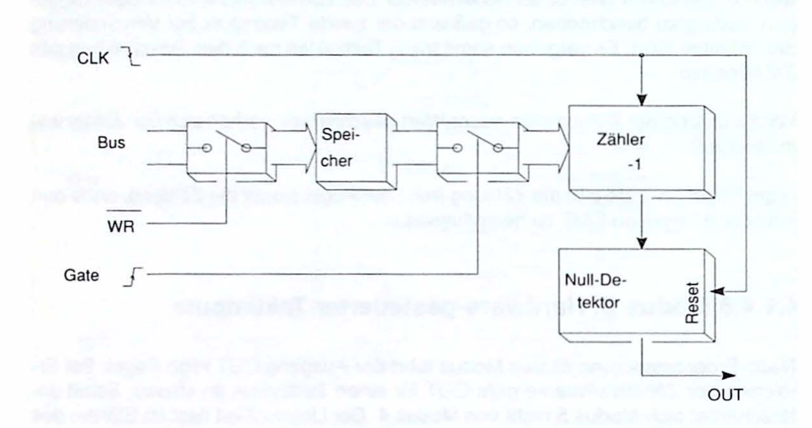
4.1.4.6 Modus 5: Hardware-gesteuerter Taktimpuls

Nach Programmierung dieses Modus führt der Ausgang OUT High-Pegel. Bei Erreichen der Zählernullmarke geht OUT für einen Taktzyklus an Masse. Somit unterscheidet sich Modus 5 nicht von Modus 4. Der Unterschied liegt im Starten des Zählers. Während es in Modus 4 durch die Software mit dem Schreiben eines Wertes erfolgt, geschieht das in Modus 5 mit der positiven Flanke am Eingang Gate. Das Beschreiben des Zählers hat nicht automatisch das Laden des Zählelementes zur Folge. Da ein Taktzyklus nach der positiven Flanke an Gate den Ladevorgang bewirkt, wird das Zählelement nach (n+1) Zählungen die Nullmarke erreichen. Werden die Zählregister zwischenzeitlich mit neuen Werten beschrieben, hat das keinen Einfluß auf den aktuellen Zählvorgang. Erst beim nächsten positiven Impuls an Gate wird der neue Zählerstand übernommen. Der Pegel am Eingang Gate hat keinen Einfluß auf den Zustand des Ausgangs OUT.

Ein Zusammenfassung des Einflusses von Gate auf das Zählverhalten zeigt Tabelle 4-2. Der logische Pegel an Gate wird jedesmal mit der positiven Flanke des CLK-Signals abgefragt. Der Eingang Gate wird in den Modi 0 und 4 durch den anliegenden Pegel gesteuert, in den Modi 1, 2, 3 und 5 ist er flankensensitiv. Eine ansteigende Flanke an Gate setzt in diesen Modi ein internes Flip-Flop. Dieses Flip-Flop wird mit der nächsten positiven Flanke des CLK-Signals abgefragt und gleichzeitig rückgesetzt. Somit wird ein Signal an Gate immer erkannt, ohne daß es so lange anliegen muß, bis die positive Flanke des CLK-Signals erscheint. In

den Modi 2 und 3 ist der Eingang Gate also sowohl pegel- als auch flankengetriggert. Stammt in diesen Modi das CLK-Signal nicht aus dem Systemtakt, sollte zur Synchronisierung unmittelbar nach einem Schreibbefehl ein Gate-Impuls folgen.

Beim Schreiben eines Kontrollwortes an einen Zähler geht sein Ausgang OUT sofort in den für den betreffenden Modus definierten Ausgangszustand, ohne daß es eines CLK-Impulses bedürfe.



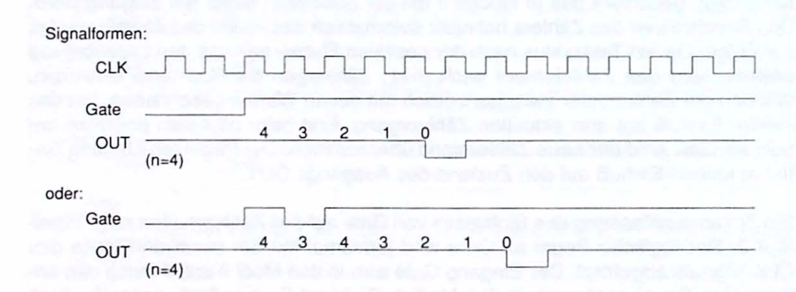


Bild 4-11. Zähler des 8253 in Modus 5

Zählermodus	Low-Pegel	positive Flanke	High-Pegel
0	Zähler steht.	------	Zähler läuft.
1	Zähler läuft.	Startet Zählvorgang und setzt OUT nach dem nächsten Takt an Masse.	Zähler läuft.
2	Zähler steht. OUT geht an Plus.	Zähler wird geladen und die Zählung beginnt.	Zähler läuft.
3	Zähler steht. OUT geht an Plus. lung beginnt.	Zähler wird geladen und die Zäh-	Zähler läuft.
4	Zähler steht.	------	Zähler läuft.
5	Zähler läuft.	Zähler wird geladen und die Zählung beginnt.	Zähler läuft.

Tabelle 4-2. Zählersteuerung durch Gate

Tabelle 4-3 zeigt die maximalen und minimalen Zählervoreinstellungen in den verschiedenen Modi. Ein Beschreiben mit dem Wert 0 ist identisch mit einem Zählwert von 2^{16} bei Binärzählung und 10^4 bei Dezimalzählung. Wenn der Zähler die Nullmarke erreicht hat, bleibt er nicht stehen, sondern läuft beim nächsten Taktimpuls unter und nimmt seinen höchsten Wert an: FF FF im Binärmodus, 99 99 im Dezimalmodus. Die Zählung wird von da aus fortgesetzt. Die Modi 2 und 3 sind periodisch, d.h. der Zähler wird automatisch mit dem Wert aus den Zählregistern nachgeladen und die Zählung wird an der Stelle fortgesetzt.

Zählermodus	Minimale Voreinstellung	Maximale Voreinstellung
0	1	0
1	1	0
2	2	0
3	2	0
4	1	0
5	1	0

Tabelle 4-3. Minimale und maximale Zählervoreinstellungen

4.1.5 Das Kontrollwort

Das Kontrollwort ist ein 8-Bit-Register, das beschrieben werden muß, um die einzelnen Zähler zur gewünschten Aktion zu veranlassen. Die im Kontrollwort geschriebene Information wird teilweise an die Zähler weitergereicht und dort zu deren Steuerung gespeichert. Die Nummer des gewünschten Zählers ist Inhalt des Kontrollworts. Über das Kontrollwort erfolgt also nur die Programmierung der Zähler, nicht aber das Beschreiben. Beschrieben werden die Zähler durch das entsprechende Bitmuster an den Eingängen A_0 und A_1. Um das Kontrollwort zu beschreiben, müssen A_0 und A_1 High-Pegel führen.

Nach dem Anlegen der Spannung sind der Inhalt und die Ausgänge der Zähler sowie der des Kontrollworts unbestimmt. Der 8253 muß also in jedem Fall möglichst rasch in der Reset-Routine durch Beschreiben des Kontrollworts initialisiert werden. Die CPU muß ihn wie ein I/O-Port bzw. wie ein RAM-Baustein ansprechen. Die Funktionsfähigkeit des Bausteins wird damit erst durch die Software hergestellt. Werden alle drei Zähler benutzt, muß das Kontrollwort dreimal hintereinander beschrieben werden. Erst danach können die Zähler mit den gewünschten Werten beschrieben werden.

4.1.5.1 Der Inhalt des Kontrollworts

SC1	SC0	RL1	RL0	M2	M1	M0	BCD

Bit Nr:	7	6	5	4	3	2	1	0

Mit dem Kontrollwort werden vier Informationen an den 8253 übermittelt:

1. Bit 0: BCD- (Dezimal-) oder Binärmodus
2. Bits 1-3: Moduswahl für den Zähler
3. Bits 4 und 5: Zahl der Bytes beim Beschreiben oder Lesen
4. Bits 6 und 7: Zählerwahl

Die Bits haben folgende Bedeutung:

BCD:

0	Binärzählung mit 16 Bits
1	Binärkodierter Dezimalzähler mit 4 Dekaden

M - Modus:

M2	M1	M0	Wirkung
0	0	0	Modus 0
0	0	1	Modus 1
X	1	0	Modus 2
X	1	1	Modus 3
1	0	0	Modus 4
1	0	1	Modus 5

Für Modus 2 und 3 kann das Bit M2 sowohl eine Eins als auch eine Null sein. Da jedoch künftige Produkte dieses Bit eventuell für andere Zwecke benutzen, sollte es mit Null beschrieben werden, um Kompatibilität mit nachfolgenden Bausteinen zu gewährleisten.

RL - Read/Load:

RL1	RL0	Wirkung
0	0	Speicherbefehl für einen Zählerstand
0	1	Schreiben/Lesen nur des Zähler-Low-Bytes
1	0	Schreiben/Lesen nur des Zähler-High-Bytes
1	1	Schreiben/Lesen von beiden Bytes; zuerst Low-Byte, dann High-Byte

SC - Select Counter:

SC1	SC0	Wirkung
0	0	Wahl von Zähler 0
0	1	Wahl von Zähler 1
1	0	Wahl von Zähler 2
1	1	Keine Wirkung

Um den Zähler 0 als Binärzähler mit zwei Bytes zu beschreiben und zu lesen und um ihn in Modus 0 arbeiten zu lassen, muß das Bitmuster 0011 0000 (30_h) an den 8253 gesendet werden. A_0 und A_1 führen dabei High-Pegel, um das Kontrollwort zu adressieren. Nachfolgend werden A_0 und A_1 an Masse gelegt, um den Zähler 0 zu adressieren. Mit zwei weiteren Schreibbefehlen wird der 16-Bit-Startwert in den Zähler geschrieben.

Soll Zähler 2 als Baudratengenerator und mit BCD-Werten arbeiten, muß er in Modus 2 versetzt werden. Das Kontrollwort muß zu diesem Zweck mit 1011 0101 ($B5_h$) beschrieben werden (Tabelle 4-4).

4.1.5.2 Das Beschreiben des 8253

Wie bereits erwähnt, muß jeder benutzte Zähler mit dem Modus und der gewünschten Voreinstellung beschrieben werden. Unbenutzte Zähler brauchen nicht programmiert zu werden. Obwohl die Programmierung des 8253 sehr flexibel erfolgen kann, müssen doch zwei Einschränkungen, die vor allem die Reihenfolge der Daten betreffen, beachtet werden:

1. Für jeden Zähler muß vor dem Schreiben der Zählwerte das Kontrollwort beschrieben worden sein.
2. Das Beschreiben der Zählregister muß in Übereinstimmung mit der in den Bits RL1 und RL0 festgelegten Reihenfolge sein; entweder nur High- oder Low-Byte oder beide Bytes, dann Low-Byte zuerst, gefolgt von High-Byte.

Die Zählregister können zu beliebiger Zeit neu beschrieben werden, ohne daß es einen Einfluß auf den eingestellten Modus hätte. Wenn der Zähler auf den Empfang zweier Bytes programmiert ist, darf zwischen dem ersten und zweiten Byte kein anderer Zugriff erfolgen, d.h. es darf zwischendurch nicht der Inhalt des Kontrollworts geändert oder ein anderer Zähler gelesen werden. Der 8253 wird dadurch zwar nicht zerstört, aber die Eintragungen in die aktuellen Zählregister wären unkorrekt.

Die Sequenzbeispiele für das Beschreiben in Tabelle 4-4 gehen davon aus, daß der 8253 für den Empfang von zwei Bytes vorbereitet ist und alle drei Zähler benutzt werden. Es sind vier von mehreren möglichen Kombinationen dargestellt.

1. Beispiel:		2. Beispiel:		3. Beispiel:		4. Beispiel:	
Schreiben von	Zähler	Schreiben von	Zähler	Schreiben von	Zähler	Schreiben von	Zähler
Kontrollwort	0	Kontrollwort	2	Kontrollwort	0	Kontrollwort	1
LSB	0	Kontrollwort	1	Kontrollwort	1	Kontrollwort	0
MSB	0	Kontrollwort	0	Kontrollwort	2	LSB	1
Kontrollwort	1	LSB	2	LSB	2	Kontrollwort	2
LSB	1	MSB	2	LSB	1	LSB	0
MSB	1	LSB	1	LSB	0	MSB	1
Kontrollwort	2	MSB	1	MSB	0	LSB	2
LSB	2	LSB	0	MSB	1	MSB	0
MSB	2	MSB	0	MSB	2	MSB	2

Tabelle 4-4. Beispiele für Beschreibungssequenzen

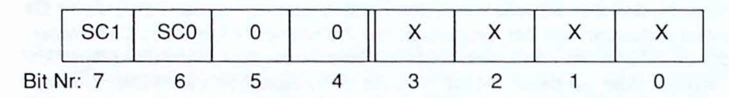
4.1.5.3 Das Lesen des 8253

In häufigen Anwendungen, besonders in Fällen externer Ereigniszählungen wird es gewünscht, daß der aktuelle Inhalt des Zählers gelesen werden kann, damit die CPU Entscheidungen auf der Grundlage der Zählwerte treffen kann. Der Zählerstand der drei Zähler im 8253 kann jederzeit gelesen werden, ohne dazu den Zähler zu stoppen oder zu beeinflussen. Für die CPU bestehen zwei Möglichkeiten, den Inhalt der Zähler zu erfahren.

Die erste Methode ist die Adressierung des betreffenden Zählers über die Eingänge A_0 und A_1, gefolgt von einem schlichten Lesebefehl. Je nach den eingestellten Werten in den Bits RL0 und RL1 des Kontrollworts für den ausgewählten Zähler liest der Prozessor nun das MSB oder das LSB. Sind beide Bits gesetzt, muß die CPU zwei Lesezugriffe ausführen, um beide Bytes zu erhalten. Das erste Byte ist das LSB, das zweite das MSB des Zählers. Wegen der internen Logik des Bausteins kann in diesem Fall nicht auf das Lesen des zweiten Bytes verzichtet werden. Es muß auch eventuell der angesprochene Zähler entweder durch Kontrollierung des Eingangs Gate oder mittels externer Logik kurz angehalten werden, um zuverlässige Zählwerte zu erhalten. Ist das Zählintervall sehr kurz und die Taktrate am Eingang CLK sehr hoch, ist in jedem Fall dazu zu raten. Bei größeren Intervallen, z.B. eine Sekunde, und langen CLK-Perioden, z.B. die Messung von Niederfrequenzen, ergeben sich keine größeren Fehler, wenn man diese Vorsichtsmaßnahme nicht beachtet.

Die zweite Methode benutzt einen Befehl an das Kontrollwort, das mit $A_1,A_0 = 11$ adressiert ist. In diesem Befehl müssen die Bits RL0 und RL1 gelöscht sein und die Bits SC1 und SC0 den gewünschten Zähler adressieren. Der Zustand der restlichen Bits spielt keine Rolle, er wird in diesem Falle keinen Einfluß auf den Zählermodus haben. Die Information von 00 in den Bits 4 und 5 unterscheiden diesen Befehl von einem Kontrollwort.

4.1.5.4 Das Kontrollwort für einen Speicherbefehl



Mit dem Schreiben dieses Befehls wird der Inhalt des Zählelementes in die Ausgabespeicher kopiert (Bild 4-5), wo er solange bleibt, bis er von der CPU gelesen (Low-Byte zuerst) wird oder bis eine Neuprogrammierung des Zählers erfolgt. Die Ausgabespeicher werden danach den augenblicklichen Inhalt des Zählelementes bis zu einem erneuten Speicherbefehl wiederspiegeln. Die Vorteile dieser Option liegen auf der Hand. Ein sauberes Lesen ohne Hardware-Zusätze und ohne Störung des Zählvorganges ist "im Flug" möglich. Des weiteren kann mit allen drei Zählern nacheinander in dieser Weise verfahren werden, wobei der Inhalt der jeweiligen Ausgabespeicher im Anschluß und mit Muße gelesen werden kann.

Wird ein zweiter Speicherbefehl für denselben Zähler gegeben, ohne daß der Inhalt zuvor gelesen wurde, wird der zweite Befehl ignoriert. Beim Lesen erscheint der Inhalt der ersten Abspeicherung.

Auch bei dieser zweiten Lesemethode muß der äußere Rahmen berücksichtigt werden, der mit den Bits RL1 und RL0 zuvor eingestellt war. Wenn nur ein Byte programmiert war, darf auch nur ein Lesezugriff erfolgen; waren zwei Bytes programmiert, müssen anschließend auch zwei gelesen werden.

4.1.6 Anwendungen

Beispiel 1: Der Systemtakt wird recht häufig als CLK-Signal verwendet, da dadurch die Frequenz eine bekannte Größe ist und außerdem nicht durch zusätzliche Komponenten erzeugt werden muß. In Fällen, in denen der Systemtakt keinen ganzzahligen Wert besitzt und in denen es auf eine exakte Zeitgebung ankommt, wird man wohl auf eine externe Taktquelle zurückgreifen. Bild 4-12 zeigt die Verwendung des Systemtaktes in verschiedenen Prozessorensystemen.

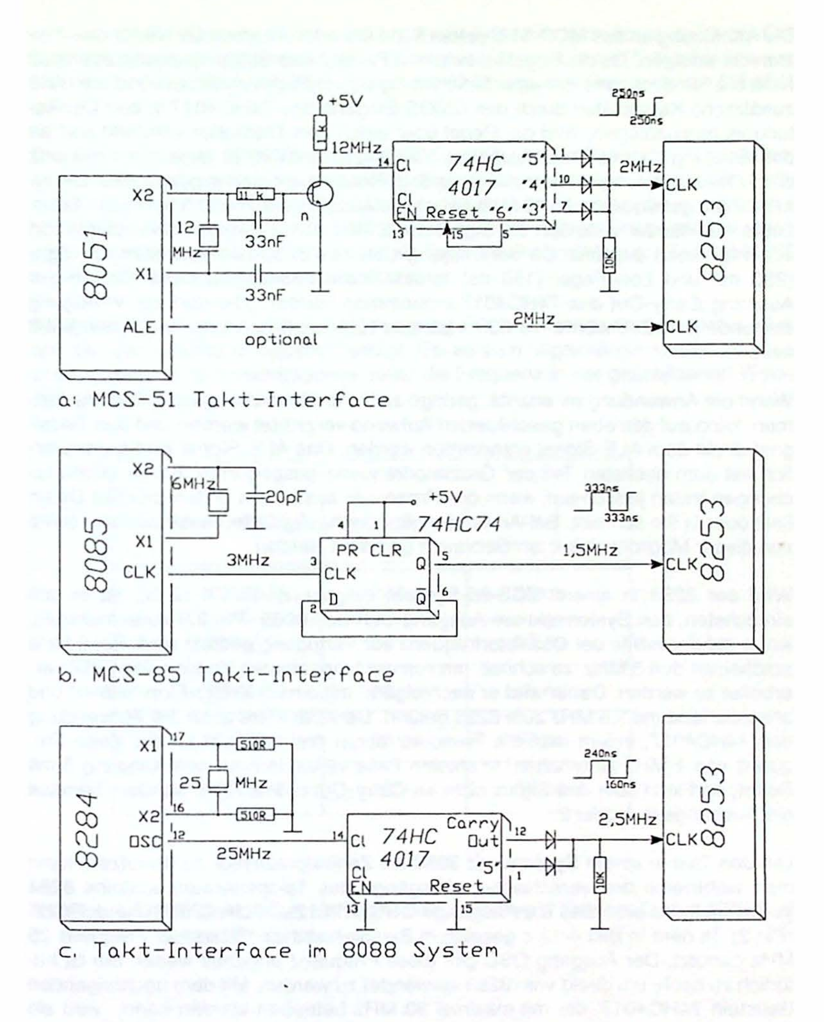


Bild 4-12. Die Verwendung des Systemtaktes zum Takten des 8253

Die Anbindung an das **MCS-51-System** kann über den internen Oszillator des Prozessors erfolgen. Da die Pegel im externen Bereich des Schwingkreises eventuell nicht die Mindestwerte der erforderlichen logischen Pegel erreichen und um nicht zusätzliche Kapazitäten durch den CMOS-Eingang des 74HC4017 in den Oszillatorkreis hinzuzufügen, wird der Pegel über einen npn-Transistor verstärkt und an den Eingang Cl des Dezimalzählers 74HC4017 geführt, in dem die Frequenz durch Rückkopplung mit dem Ausgang 6 an Reset durch sechs geteilt wird. Bei einer Eingangsfrequenz von 12 MHz geben die Ausgänge 3, 4 und 5 über eine Oder-Logik miteinander verknüpft ein Signal von 2 MHz aus, das ein Tastverhältnis von 1:1 (High:Low) aufweist. Da der Eingang CLK des 8253 Minimalzeiten für High- (230 ns) und Low-Pegel (150 ns) fordert, kann das Signal nicht einfach am Ausgang Carry-Out des 74HC4017 entnommen werden. Die dort zur Verfügung stehende Low-Zeit würde mit 83 ns zu kurz für eine Erkennung durch den 8253 sein.

Wenn die Anwendung es erlaubt, geringe zeitliche Abweichungen in Kauf zu nehmen, kann auf den eben geschilderten Aufwand verzichtet werden und das Taktsignal direkt dem ALE-Signal entnommen werden. Das ALE-Signal wird kontinuierlich mit dem sechsten Teil der Oszillatorfrequenz ausgegeben. Kurze Unterbrechungen treten jedoch auf, wenn der Prozessor aus einem Datenspeicher Daten liest oder in ihn schreibt. Bei Anwendungen, die häufig Daten austauschen, sollte von dieser Möglichkeit keinen Gebrauch gemacht werden.

Wird der 8253 in einem **MCS-85-System** eingesetzt (Bild 4-12 b), ist es am einfachsten, den Systemtakt am Ausgang CLK des 8085 (Pin 37) zu entnehmen, wo er mit der Hälfte der Oszillatorfrequenz zur Verfügung gestellt wird. Bei 6 MHz erscheinen dort 3 MHz, zu schnell, um von der langsamsten Version des 8253 verarbeitet zu werden. Daher wird er nachfolgend mit einem D-Flip-Flop halbiert und anschließend mit 1,5 MHz zum 8253 geführt. Denkbar wäre auch die Verwendung des 74HC4017, indem man die Frequenz durch drei teilt, um so die glatte Frequenz von 1 MHz zu erhalten. In diesem Falle verbinde man den Ausgang 3 mit Reset, darf jetzt aber das Signal nicht an Carry-Out entnehmen, sondern benutze die Ausgänge 0, 1 oder 2.

Um den Takt in einem **System mit 8088** als Zentralprozessor zu benutzen, kann man wahlweise drei verschiedene Ausgänge des Taktgeneratorbausteins 8284 verwenden. Es sind dies die Ausgänge OSC (Pin 12), CLK (Pin 8) und PCLK (Pin 2). In dem in Bild 4-12 c gezeigtem Beispiel wird ein Quarz der Frequenz 25 MHz benutzt. Der Ausgang OSC gibt diese Frequenz ungeteilt weiter. Sie ist natürlich zu hoch, um direkt vom 8253 verwendet zu werden. Mit dem nachfolgenden Baustein 74HC4017, der mit maximal 30 MHz betrieben werden kann, wird sie durch zehn geteilt und das Signal mit einem Tastverhältnis von 1:1 an Carry-Out ausgegeben, d.h. High- und Low-Pegel weisen eine Zeitdauer von 200 ns auf, ausreichend für den Low-Pegel, für den High-Pegel aber zu kurz. Aus diesem Grund

wird der High-Pegel des Ausgangs 5 mit der Zeitdauer von 40 ns zu Carry-Out über eine Diodenlogik hinzuaddiert. Daraus resultiert nun ein Zeitverhältnis von 240 ns zu 160 ns, und am Eingang CLK des 8253 liegt eine Frequenz von 2,5 MHz an.

Das Signal, das der 8284 am Ausgang CLK ausgibt, ist ein Drittel der Oszillatorfrequenz. Durch diese Teilung ergibt sich eine nicht ganzzahlige Taktfrequenz, die nur mit zusätzlichem Software-Aufwand zu handhaben ist, es sei denn, man wählt eine für die Teilung geeignete Quarzfrequenz. Auch dieses Signal muß nachfolgend durch vier geteilt werden, wobei als Teiler ein Binärzähler vom Typ 74HC393 oder 74HC4024 verwendet werden kann.

Auf gleiche Weise ist das Signal am Ausgang PCLK zu verwenden, das den sechsten Teil der Oszillatorfrequenz beträgt. Es wird im allgemeinen immer noch zu schnell sein, so daß nachfolgende Teiler die Frequenz in der geschilderten Weise reduzieren müssen.

Bausteinnummer	High	Low
8253(-5)	230 ns	150 ns
8254(-5)	60 ns	60 ns
8254-2	30 ns	50 ns
82C54	60 ns	60 ns
82C54-2	30 ns	50 ns

Tabelle 4-5. Mindestzeiten für High- und Low-Pegel für den Zähler 8253 und seine Verwandten

Beispiel 2: Das nachfolgende Beispiel (Bild 4-13) schildert den Einsatz des 8253 als externer Ereigniszähler auf Sekundenbasis. Zähler 0 und Zähler 1 erzeugen zusammen mit einem 1-MHz-Signal die nötige Zeitbasis und stellen gleichzeitig dem System wichtige Zeiten zur Verfügung, die für eine Interrupt-Steuerung benutzt werden können. Das externe Ereignis wird in Form einer von einer analogen Größe abhängigen Frequenz gebildet, die Zähler 2 zugeführt wird. Nach einer Sekunde geht der Ausgang OUT des Zählers 1 an Masse, löst am System einen Interrupt aus, verhindert durch Low-Pegel an den Eingängen Gate von Zähler 0 und 2 weitere Zählungen. Zeitbasis und Ereigniszähler sind gestoppt. Die CPU hat nun Zeit, um in aller Ruhe Zähler 2 auszulesen. Um den Vorgang erneut zu starten, muß lediglich ein Schreibbefehl an Zähler 1 erfolgen. In diesem Beispiel setzt das Beschreiben mit dem Wert 0100_h den Ausgang OUT an Plus und gibt nun Zähler 0 und 2 für weitere Zählungen frei. Mit der positiven Flanke an Gate werden deren voreingestellten Werte automatisch nachgeladen, so daß ein Schreiben an die beiden Zähler nicht nötig ist.

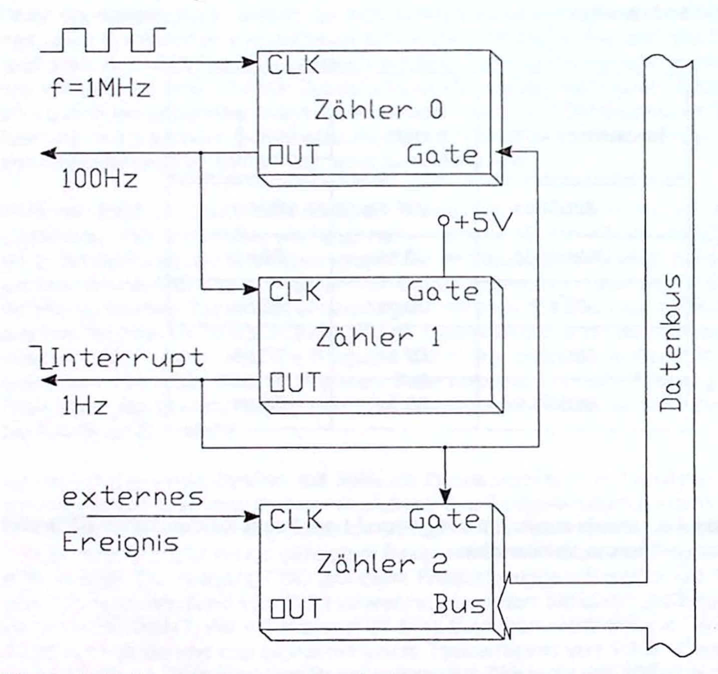


Bild 4-13. Selbstanhaltender Ereigniszähler

Alle drei Zähler werden mit Modus 2 voreingestellt. Zähler 0 und Zähler 1 arbeiten im BCD-Modus, wohingegen der Modus von Zähler 2 abhängt von der Zählrate oder der Software, die die Zahlen weiterverarbeiten soll. Zähler 0 und 2 werden in der Reset-Routine mit dem Wert 00 00, Zähler 1 mit dem Wert 01 00_h voreingestellt. Somit teilt Zähler 0 das 1-MHz-Signal durch 10^4 und gibt ein 100-Hz-Signal aus, das gleichzeitig Zähler 1 taktet. Zähler 1 teilt die Taktrate durch 10^2 und erzeugt das 1-Hz-Signal. Da die Zähler Abwärtszähler sind, ist der Zählstand in Zähler 2 nicht identisch mit der Zahl der Ereignisse. Die CPU muß den Binärwert komplementieren und anschließend inkrementieren, um den reellen Wert zu erhalten.

Werden beispielsweise in einer Sekunde 2118 Ereignisse registriert, liest die CPU den Wert F7 BE_h aus dem Zähler 2, der nach Komplement- und Inkrementbildung zu 08 46_h = 2118_d wird.

Eine einfache Weise, den Baustein 8253 zu testen, steht den Besitzern eines PCs zur Verfügung. Er ist dort ein fester Bestandteil der Hardware und ist unter den I/O-Adressen 40 bis 43 ansprechbar. Zähler 0 und Zähler 1 werden für die Systemsteuerung benötigt und sollten, um einen Absturz des Rechners zu vermeiden, unangetastet bleiben. Lediglich ein Auslesen des aktuellen Zählerstandes ist möglich.

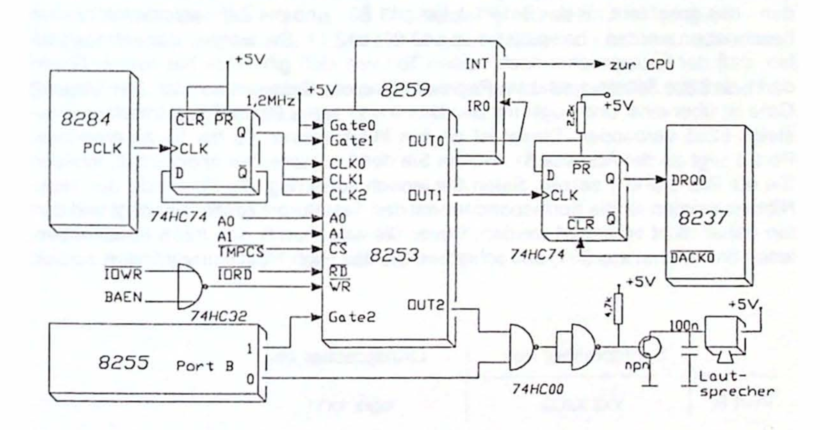


Bild 4-14. Der Zähler 8253 im 8088-System

Alle drei Zähler werden mit dem Peripherietaktsignal PCLK des 8284 versorgt, das jedoch zuvor noch durch zwei geteilt werden muß. Somit steht an den Takteingängen CLK des 8253 eine Frequenz von 1,2 MHz an. Der Ausgang OUT von Zähler 0 ist an den Interrupt-Eingang 0 des Interruptcontrollers 8259 geführt und erzeugt über diesen Baustein 18,2 mal in der Sekunde den Timerinterrupt in der CPU. Das Signal von Zähler 1 dient zum Takten eines D-Flip-Flops, dessen Ausgang ein Direktspeicherzugriff DREQ0 am DMA-Controller 8237 auslöst (Bild 4-14).

Zähler 2 wird mit derselben Frequenz versorgt und nur gelegentlich vom System benötigt, nämlich genau dann, wenn über den eingebauten Lautsprecher Töne ausgegeben werden sollen. Der Ausgang OUT führt zur Basis eines Schalttransistors, der den Strom für den Lautsprecher freigibt. Somit kann man Zähler 2 zu eigenen Experimenten nutzen.

Den einfachsten Zugriff auf den 8253 stellt das Programm DEBUG mit den Befehlen IN (i) und OUT (o) zur Verfügung. Die Inhalte der Zähler 0, 1 bzw. 2 erscheinen an den Adressen 40, 41 bzw. 42, das Kontrollwort wird über Adresse 43 beschrieben. Beim Auslesen des Zählers 2 mit der Befehlsfolge i42; i42 kann man die letzte Eintragung des ROM-BIOS in der Reihenfolge Low-Byte/High-Byte erfahren. Sie können somit direkt die Wirkung kontrollieren, die der BASIC-Befehl SOUND auf den Zähler 2 hat. Zur Erzeugung von Tönen muß Zähler 2 in Modus 3 versetzt werden - das geschieht mit der Befehlsfolge o43 B6 - und die Zählregister mit Werten beschrieben werden - beispielsweise o42 00; o42 11. Sie werden danach feststellen, daß der Lautsprecher noch keinen Ton von sich gibt. Das hat seinen Grund darin, daß der Zähler durch Low-Pegel am Eingang Gate gesperrt ist. Der Eingang Gate ist über eine *Und*-Logik mit den Bits 0 und 1 des Ports B des Interface-Bausteins 8255 verbunden. Dieser ist an den I/O-Adressen 60 bis 63 zu erreichen. Port B liegt an der Adresse 61. Wollen Sie den Lautsprecher anschalten, müssen Sie die Bits 0 und 1 setzen. Seien Sie jedoch vorsichtig, die Zustände des High-Nibbles werden für die Korrespondenz mit den Tastaturprozessor benötigt und dürfen daher nicht verändert werden. Bevor Sie also Port B des 8255 beschreiben, lesen Sie ihn erst aus (i61) und schreiben Sie das High-Nibble unverändert zurück:

	Lautsprecher aus	Lautsprecher ein
Port B:	xxx xx00	xxxx xx11

Ergibt ein Auslesen des Ports B beispielsweise den Wert 58, muß zum Einschalten des Lautsprechers der Wert 5B (o61 5B) zurückgeschrieben werden. Ein Beschreiben mit dem alten Wert bewirkt ein Ausschalten.

Vom Timer-Baustein 8253 gibt es einige Verwandte, die sich nur geringfügig vom Standardbaustein unterscheiden. Die wichtigsten Unterschiede zeigt Tabelle 4-6.

Bausteinnummer	Zugriffszeit	Ausführung	Taktrate	Stromaufnahme
8253	400 ns	NMOS	2,6 MHz	140 mA
8253-5	300 ns	NMOS	2,6 MHz	140 mA
8253-2	200 ns	NMOS	5 MHz	140 mA
8254-5	150 ns	NMOS	5 MHz	140 mA
8254	150 ns	NMOS	8 MHz	140 mA
8254-2	95 ns	NMOS	10 MHz	140 mA
82C54	150 ns	CMOS	8 MHz	10 mA
82C54-2	95 ns	CMOS	10 MHz	10 mA

Tabelle 4-6. Die wichtigsten technischen Daten für den Zähler 8253 und seine Verwandten.

4.2 Die programmierbaren Intervall-Timer-Bausteine 8254 und 82C54

4.2.1 Unterschiede zum Baustein 8253

Die Timer 8254 und 82C54 sind exakte Kopien das Bausteins 8253 mit zusätzlichen Eigenschaften und sind somit pin- und abwärtskompatibel. Ein 8253 kann also in jedem Fall durch einen 8254 ersetzt werden, aber ein 8254 nicht unbedingt durch einen 8253. Die Unterschiede sind sehr gering und beschränken sich im wesentlichen auf das Auslesen eines Zählerstatusregisters, das im 8253 nicht vorhanden ist. Verzichtet die Software auf diese Option, ist der 8254 dem 8253 völlig gleichwertig.

Der 82C54 ist lediglich die CMOS-Version des 8254 und verfügt über keine zusätzlichen Eigenschaften.

Die vorausgehende Beschreibung des 8253 hat ohne Einschränkung auch seine Gültigkeit für den 8254. Daher werden im folgenden auch nur die zusätzlichen Eigenschaften genannt.

Unterschiede:

1. Das Beschreiben und Lesen eines Zählers kann abwechselnd erfolgen. Ein Beispiel zeigt diese Eigenschaft: Der Zähler sei für das Zählen von zwei Bytes programmiert. So kann nach dem Lesen des LSB des Zählers unmittelbar der neue Wert für das LSB geschrieben werden. Danach wird das MSB gelesen und dann erneut beschrieben. Diese Unterbrechung des Auslesens ist im 8254 nicht möglich und ist sicher kein großer Nachteil. Es ist nur darauf zu achten, daß zwischen den Bytes keine Neubeschreibung des Kontrollworts erfolgt.

2. Bedeutsamer ist der Read-Back-Befehl. Der Read-Back-Befehl ist möglich, weil ein Zusatz in der Hardware gewisse Informationen aus der Kontrollogik und den Zustand des Ausgangs Out zwischenspeichert und ihn nach einem entsprechenden Befehl über den internen Bus ausgibt (Bild 4-15).

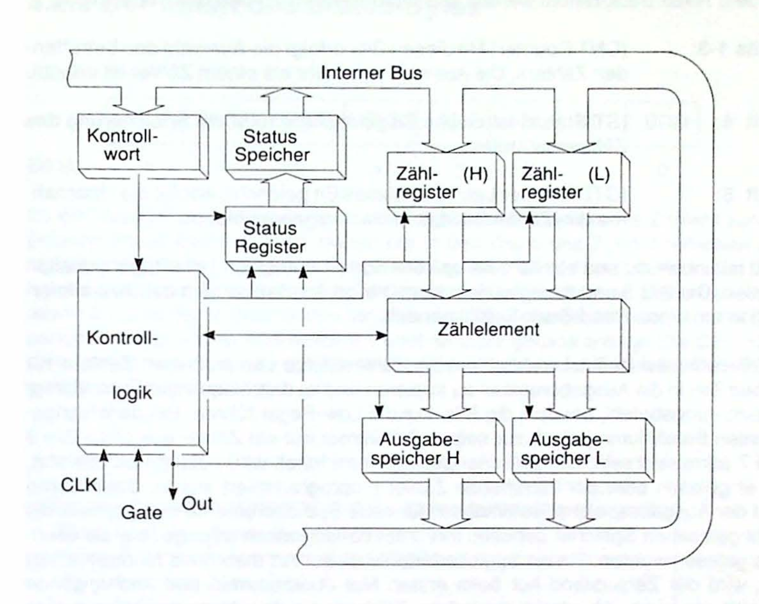
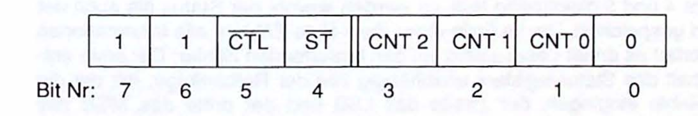


Bild 4-15. Der interne Zähler im 8254

Der Read-Back-Befehl unterscheidet sich von allen anderen Befehlen durch Einsen in den Bits 6 und 7. Er wird in das Kontrollwort geschrieben.

4.2.2 Das Read-Back-Befehlsformat im 8254



	1	1	$\overline{CTL}$	$\overline{ST}$	CNT 2	CNT 1	CNT 0	0
Bit Nr:	7	6	5	4	3	2	1	0

Mit dem Read-Back-Befehl werden drei Informationen an den 8254 übermittelt:

Bits 1-3: (CNT Counter) Mit diesen Bits erfolgt die Auswahl des betreffenden Zählers. Die Auswahl von mehr als einem Zähler ist erlaubt.

Bit 4: (ST Status) Ist dieses Bit gelöscht, erfolgt die Speicherung des Zählerzustandes.

Bit 5: (CTL Counter Latch) Ist dieses Bit gelöscht, erfolgt die Übernahme des Zählerstandes in die Ausgabespeicher.

Bit 0 ist unbenutzt und soll für eine spätere Kompatibilität auf Low-Pegel gehalten werden. Die Bits 5 und 6 zeigen dem Kontrollwort an, daß es sich bei dieser Information um einen Read-Back-Befehl handelt.

Der Befehl kann benutzt werden, um die Zählerstände von mehreren Zählern zur selben Zeit in die Ausgaberegister zu kopieren und ist damit äquivalent zum Zähler Speicherungsbefehl, bei dem die Bits 4 und 5 Low-Pegel führen. Bei dem letztgenannten Befehl kann jedoch zur selben Zeit immer nur ein Zähler durch die Bits 6 und 7 adressiert sein. Der zwischengespeicherte Inhalt wird solange aufbewahrt, bis er gelesen oder der betreffende Zähler umprogrammiert wurde. Beim Lesen wird der Ausgabespeicher automatisch für neue Speicherbefehle freigegeben, die nicht gelesenen Speicher behalten ihre Information jedoch solange, bis sie ebenfalls gelesen wurden. Treten Speicherbefehle dieser Art mehrmals hintereinander auf, wird der Zählerstand nur beim ersten Mal übernommen und nachfolgende Befehle ignoriert, d.h. der aufbewahrte Zählwert ist der Wert, der beim ersten Speicherbefehl übernommen wurde.

Wahlweise können mit dem Read-Back-Befehl gewisse Zustände im Innern des Zählers abgefragt und von der CPU ausgewertet werden. Das ist nützlich, wenn beispielsweise der logische Pegel des Ausgangs OUT von Interesse ist. Eine sonst notwendige Hardware-Verbindung entfällt dadurch. Damit der Status gelesen werden kann, muß er zuvor mit Bit 4 gleich Null gespeichert werden. Auch hier kann der Status mehrerer Zähler gleichzeitig aufgefangen werden. Um die erneute Statusspeicherung freizugeben, muß hier wie bei den Zählinhalten zunächst ein Lesezugriff erfolgen.

Sind die Bits 4 und 5 gleichzeitig Null, so werden sowohl der Status als auch der Zählerstand gespeichert. Um im Falle eines Zwei-Byte-Zählers alle Informationen zu lesen, bedarf es dreier Lesezugriffe auf den betreffenden Zähler: Der erste enthält den Inhalt des Statusregisters unabhängig von der Reihenfolge, mit der die Speicherbefehle eingingen, der zweite das LSB und der dritte das MSB des Zählers.

4.2.3 Der Inhalt des Status-Bytes

	OUT	NULL	RL1	RL0	M2	M1	M0	BCD
Bit Nr:	7	6	5	4	3	2	1	0

Es wird also im wesentlichen der zuvor programmierte Modus des Zählers zurückgelesen. Neue Informationen stehen nur in den Bits 6 und 7. Bit 7 reflektiert den logischen Pegel des Ausgangs Out, Bit 6 zeigt mit einer Null an, ob der in den Zähler geschriebene Zählwert in das Zählelement übernommen ist. Denn das hängt davon ab, welcher Modus für den betreffenden Zähler eingestellt war, ob eine Taktperiode bereits um ist und welcher Pegel an Gate gerade anliegt. Die CPU kann daran erkennen, ob der ausgelesene Wert der aktuelle oder Datenabfall von übergangenen Ereignissen ist.

Vorgang	NULL-Bit
Schreiben des Kontrollworts	1
Schreiben des Zählwertes	1
Übernahme des Zählwertes in das Zählelement	0

Tabelle 4-7. Die Beeinflussung des NULL Bits

Beispiele für mögliche Read-Back-Befehlssequenzen gibt Tabelle 4-8.

Bitinformationen	Befehl für:	gespeichert wird:
1100 0010	Zähler- und Statusspeicherung von Zähler 0	Zählerstand und Status des Zählers 0
1110 0100	Statusspeicherung von Zähler 1	Status von Zähler 1
1110 1100	Statusspeicherung der Zähler 2 und 1	Status nur von Zähler 2
1101 1000	Zählerspeicherung von Zähler 2	Zählerstand von Zähler 2
1100 0100	Zähler und Statusspeicherung von Zähler 1	Zählerstand von Zähler 1, aber kein Status
1110 0010	Statusspeicherung von Zähler 1	Keine Wirkung, da Status bereits gespeichert

Tabelle 4-8. Beispiel für mögliche Read-Back-Sequenzen

4.3 Der programmierbare DMA-Controller 8237

4.3.1 Der DMA-Transfer

Die drei Buchstaben DMA stehen für Direct Memory Access, was mit direktem Speicherzugriff zu übersetzen ist. Da diese drei Buchstaben mittlerweile einen Terminus technicus darstellen, werden sie im folgenden nicht übersetzt, sondern unverändert in diesem Sinne verwendet.

Der Sinn und damit der Nutzen eines Direktspeicherzugriffs DMA ist zu erkennen, wenn man sich in Erinnerung ruft, wie die CPU im System Daten von einem Platz zu einem anderen Platz schiebt. Das ist ein Vorgang, wie er sehr häufig bei der Kommunikation von Prozessor und Peripherie auftritt. Wird zum Beispiel ein Programm von der Diskette in den Arbeitsspeicher geladen, so müssen sehr viele Daten in der gleichen Reihenfolge in den RAM-Bereich kopiert werden. Der Floppy-Disk-Controller steuert dabei die Mechanik des Laufwerks und stellt das gerade von der Diskette gelesene Byte zur Verfügung. Er müßte bei der herkömmlichen Weise der CPU eine Mitteilung machen, z.B. über einen Interrupt, daß sie die anstehenden Daten lesen kann. Die CPU müßte nun mittels Lesebefehl das Byte in den Akkumulator des Prozessors laden, die gewünschte Zieladresse suchen oder erzeugen und mittels Schreibbefehl den Akkumulatorinhalt an die RAM-Adresse schreiben. Für eine solche Datenbewegung sind mindestens drei Maschinenbefehle notwendig. Betrachtet man sich den Zeitbedarf für einen derartigen Transfer, so sieht man, wie langsam der Vorgang abläuft. Die Prozessoren 8086 bis 80386 beispielsweise investieren allein für die beiden Schreib-/Lesebefehle 16 - 20 Taktzyklen, für die Adressenaktualisierung und Schleifenbildung nochmals 20 Taktzyklen. Der Vorgang wird langsam und durch die Erstellung der notwendigen Software umständlich. Ein ähnlicher Fall tritt auf, wenn über die serielle Schnittstelle Daten zu senden oder zu empfangen sind.

Oftmals sind auch größere Datenmengen im RAM selbst zu kopieren. Wenn ein Bildschirminhalt durch einen anderen ersetzt werden soll, können im RAM die Bildschirmaufbauten bereits vorgefertigt vorliegen und brauchen bei Bedarf nur in den Bildschirmspeicher kopiert zu werden.

Wie auch immer, bei all diesen Datenwanderungen, ist die CPU das zeitbestimmende Mittelglied.

Ein DMA-Controller nun übernimmt genau diese Aufgaben und erledigt sie in wesentlich kürzerer Zeit. Am einfachsten wäre es, wenn Datenquelle und Datenziel über den Datenbus miteinander verbunden wären und gleichzeitig an die Quelle ein Lese- und an das Ziel ein Schreibimpuls ausgehen würde. Leider ist das nicht immer möglich, da die Quelle oder/und das Ziel adressiert werden müssen und beide den Adreßbus gemeinsam haben. Das ist nur im Falle einer RAM-I/O-Kombination möglich, wenn der RAM-Bereich im Vergleich zum I/O-Bereich über getrennte Adreßleitungen verfügt.

Im Falle des Speicher-Speicher-Transfers bleibt dem DMA-Controller somit nichts anderes übrig, als ähnlich der CPU den Quellinhalt zunächst zu lesen, ihn intern zwischenzuspeichern und ihn anschließend in das Ziel zu schreiben. Der Unterschied zum Weg über die CPU liegt in der Geschwindigkeit. Der DMA-Controller benötigt für diesen Vorgang drei oder vier Taktzyklen, die CPU hingegen mindestens 40. Die CPU teilt dem DMA-Controller zu Beginn des Transfers die Quelle, die Zahl der zu übertragenden Bytes und das Ziel mit. Anschließend übergibt sie dem DMA-Controller die Kontrolle über den Adreß- und Datenbus. Während sich der DMA-Controller mit seiner Arbeit sputet, kann die CPU leider nicht weiterarbeiten, da ja Daten- und Adreßbus vom DMA-Controller benötigt werden. Sie geht solange in den Halt-Zustand oder Wait-State.

Die wenigen genannten Beispiele zeigen bereits, daß es wünschenswert ist, wenn man die Art des DMA-Transfers auf die augenblickliche Situation einstellen kann. Häufig auftretende Situationen sind der Transfer vom Speicher an die Peripherie (RAM—>I/O), z.B. das Sichern von Daten auf Diskette, oder der umgekehrte Vorgang, das Einlesen von Daten (I/O—>RAM), oder das Vergleichen (Verifizieren) von Daten, oder der Speicher-zu-Speicher-Transfer (RAM—>RAM). Darüber hinaus kann es wünschenswert sein, wenn in jeder Art der Datenübertragung die Möglichkeit besteht, einzelne Bytes oder ganze Blocks zu kopieren, den Kopiervorgang zu unterbrechen, vorzeitig zu beenden, schneller oder langsamer zu gestalten, um der Geschwindigkeit des Speichers gerecht zu werden etc. Das alles leistet kein von der Hardware festeingestellter Logikbaustein, sondern lediglich ein programmierbarer Chip, in diesem Fall ein DMA-Controller.

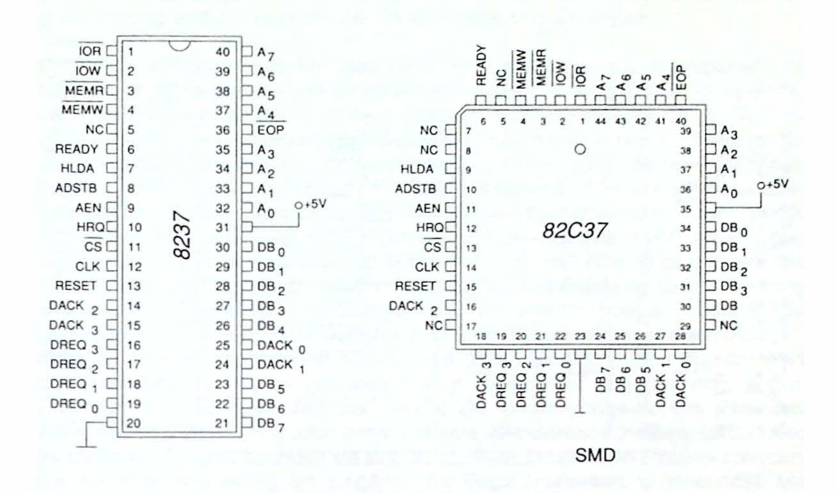


Bild 4-16. Pin-Belegung des DMA-Controllers 8237

4.3.2 Übersicht

Der DMA-Controller 8237 ist ein Baustein, der mit allen Mikrocontrollern und Prozessoren der 80XX-Reihe zusammenarbeiten kann. Er wird mit demselben Signal getaktet, mit dem die CPU getaktet wird, und bewältigt bei 8 MHz eine Übertragungsrate von ungefähr 4 MByte pro Sekunde (I/O-Operation).

Damit er seine Aufgabe im System erfüllen kann, muß er initialisiert und bei Bedarf neu programmiert werden. Zum Beschreiben und Lesen dienen die Anschlüsse DB_{0-7}, über sie fließen die Daten. Ein Schreibimpuls wird dem Pin IOW zugeführt, ein Leseimpuls dem Pin IOR. Die Wahl des gewünschten internen Registers erfolgt zusammen mit Chip-Select $\overline{CS}$ über die Register-Select-Inputs A_{0-3} und erfolgt in der üblichen Manier, mit der ein I/O-Baustein beschrieben oder gelesen wird. Die ebengenannten Pins besitzen diese Funktion nur, wenn der Baustein keinen DMA ausführt. Bei einem DMA werden sie für andere Zwecke verwendet.

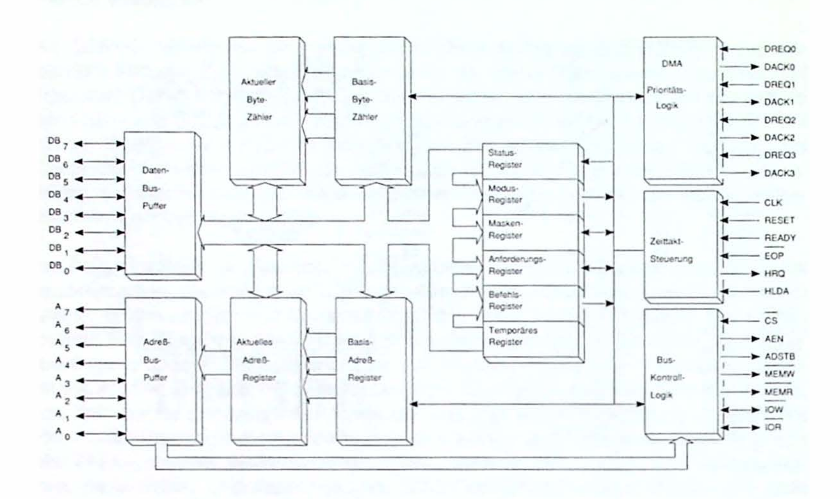


Bild 4-17. Blockdiagramm des DMA-Controllers 8237

Zur DMA-Aktivierung muß dem 8237 ein entsprechendes Signal zugeführt werden, das von dem Baustein stammt, der eine Datenübertragung wünscht. Das muß nicht unbedingt die CPU sein, das kann auch von einem UART oder einem Floppy-Disk-Controller stammen. Insgesamt verfügt der 8237 über vier Eingänge, über die ein DMA-Wunsch von vier verschiedenen Quellen angemeldet werden kann; sie sind mit DREQ (DMA Request) bezeichnet. Um dem anfordernden Gerät mitzuteilen, daß und wie lange der 8237 mit dem DMA-Transfer beschäftigt ist, besitzt der DMA-Controller die Ausgänge DACK (DMA Acknowledge). Die Art des aktiven Pegels dieser Ein- und Ausgänge kann programmiert werden, um ihn zur Zusammenarbeit mit den unterschiedlichsten Systemen zu befähigen.

Nach der Anmeldung eines DMA-Wunsches kann der DMA-Controller nicht sofort aktiv werden, da der Bus im allgemeinen noch durch den laufenden Befehl der CPU belegt ist. Zuvor muß der Transfermodus, die Startadresse und die Zahl der zu übertragenden Bytes in den DMA-Controller geschrieben werden. Die CPU erfährt die Absicht des DMA-Controllers, den Bus zu übernehmen, durch High-Pegel auf der Leitung HRQ (Hold Request). Sobald die CPU ihre Aktivität angehalten hat,

wird das dem DMA-Controller über den Eingang HLDA (Hold Acknowledge) mitgeteilt. Die nachfolgenden Aktionen differieren untereinander, da sie von der Programmierung und der Herkunft der DMA-Anforderung abhängen.

Der DMA-Controller ist in der Lage, über 16 Pins eine Adresse auszugeben und kann somit einen Bereich von 64 KByte adressieren. Das Low-Byte der Speicheradresse erscheint an den Pins A_{0-7}, das High-Byte an den Pins DB_{0-7}. Da kurz nach Ausgabe des Adressen-High-Bytes die Pins wieder in den Tri-State gehen und eventuell (Speicher-zu-Speicher-Modus) Daten über diese Anschlüsse fließen, muß das High-Byte in einem externen Adreßzwischenspeicher aufgefangen werden. Zu diesem Zweck besitzt der DMA-Controller die Ausgänge AEN (Address Enable) und ADSTB (Address Strobe). Mit High-Pegel am Ausgang AEN wird der Tri-State Adreßpuffer aktiviert und mit dem ADSTB-Signal wird das High-Byte in den Puffer getaktet. Somit hat das ADSTB-Signal die gleiche Wirkung wie das ALE-Signal des Prozessors bzw. des Bus-Controllers. Während die beiden letztgenannten kontinuierlich ein ALE-Signal ausgeben, geschieht das im Falle des DMA-Controllers mit dem ADSTB-Signal nur dann, wenn sich der Wert des Adressen-High-Bytes geändert hat, in jedem Fall aber bei dem ersten DMA-Zyklus. Das spart Zeit und erhöht die Geschwindigkeit. Die Pins des Adressen-Low-Bytes A_{0-7} sind direkt mit dem dekodierten Adreßbus verbunden. Im inaktiven Zustand befinden sie sich im Tri-State, lediglich die Pins A_{0-3} werden bei der Programmierung als Eingänge zur Registerselektierung verwendet. Mit den vier Steuerleitungen $\overline{MEMW}$, $\overline{MEMR}$, $\overline{IOW}$ und $\overline{IOR}$, die gemäß des programmierten Transfertyps aktiv werden, erfolgt nun die Datenübertragung. Ist sie für einen Speicher- oder I/O-Baustein zu schnell, kann mittels Eingang Ready eine Verzögerung bewirkt werden. Bei der Datenübertragung von Speicher zu Speicher sind besondere Bedingungen zu beachten, da in diesem Fall die Quell- und Zieladresse differieren.

Die genaue Funktion der einzelnen Pins wird zusammenfassend in Tabelle 4-9 beschrieben.

4.3.3 Die Funktionen der Pins

Symbol	Pin-Nummer	Typ	Beschreibung
+5 V, Masse	31, 20	I I	Stromversorgungsanschlüsse. Für eine zuverlässige Entkopplung sollte zwischen die Pins 31 und 20 ein 0,1- µF-Kondensator geschaltet werden.
CLK	12	I	Clock. Das Signal am Takteingang wird zur Zeittaktsteuerung der Operationen des DMA-Controllers benötigt. Die Maximalfrequenz kann je nach Bausteintypzwischen 3,1 MHz und 10 MHz liegen.
$\overline{CS}$	11	I	Chip Select. Mit Low-Pegel an diesem Eingang wird derBaustein für die Kommunikation mit der CPU freigegeben. Liegt der Eingang HLDA an High-Pegel, wird derEingang $\overline{CS}$ abgeschaltet, um versehentliches Lesen oder Beschreiben des 8237 zu verhindern.
RESET	13	I	Reset. Der Eingang dient zur Initialisierung des Bausteins und soll verhindern, daß der DMA-Controller nach Anlegen der Spannung ungewollte Aktivitäten entfaltet. High-Pegel löscht die Befehls-, Status-, Anforderungs- und temporären Register, ferner die internen Flip-Flops und den Modus-Registerzähler. Die Maskenregister werden so beschrieben, daß DMA-Anforderungen ignoriert werden.
READY	6	I	Ready. Mit Hilfe des Eingangs können Schreib- oder Lesesignale verlängert werden. Dadurch kann der DMA-Transfer langsameren Speicher- oder Peripheriebausteinen angepaßt werden. Solange Masse am Eingang anliegt, wird die DMA-Periode verlängert. Benötigt man keine Zeitverzögerung, ist der Pin mit +5 V zu verbinden.

HLDA	7	I	Hold Acknowledge. Mit High-Pegel an diesem Pin zeigt die CPU dem DMA-Controller an, daß sie ihre Aktivitäten auf dem Bus beendet hat. Es ist das Startsignal für den DMA-Beginn. Der High-Pegel bewirkt ferner, daß Signale am Eingang $\overline{CS}$ ignoriert werden, um unerwartete Schreib-/Leseoperationen auszuschließen. Zwischen der positiven Flanke des HRQ-Signals und der positiven Flanke des HLDA-Signals muß mindestens eine Taktperiode liegen.
DREQ0 DREQ1 DREQ2 DREQ3	19 18 17 16	I I I I	DMA Request. Über diese Eingänge kann ein Peripheriebaustein oder der Prozessor DMA-Transfer anmelden. Der Pegel, mit dem diese Anmeldung erfolgt, ist programmierbar. Dadurch kann der DMA-Controller mit den unterschiedlichsten Bausteinen zusammenarbeiten. Nach einem Reset sind die Eingänge high-aktiv. Das Anforderungssignal sollte solange anliegen, bis der DMA Controller über die Leitung DACK den Beginn des DMAs quittiert hat. Treten an den Eingängen DREQ gleichzeitig mehrere aktive Signale auf, so wird der Eingang 0 zuerst, der Eingang 3 zuletzt bedient. Diese Priorität kann jedoch geändert werden.
DB0 DB1 DB2 DB3 DB4 DB5 DB6 DB7	30 29 28 27 26 23 22 21	I/O I/O I/O I/O I/O I/O I/O I/O	Data Bus. Die Datenbusleitungen sind bidirektionale Tri-State-Leitungen, die mit dem Datenbus des Systems verbunden werden. Sie üben drei Funktionen aus: 1. Im Nicht-DMA-Modus werden über diese Anschlüsse die Register beschrieben oder gelesen. 2. Im DMA-Modus erscheint kurzfristig das Adressen-High-Byte, das mittels ADSTB-Signal in einem externen Adreßzwischenspeicher aufgefangen werden muß, da es sonst zu Konflikten auf dem Datenbus kommt. Danach gehen diese Leitungen in den Tri-State. 3. Bei einem Speicher-zu-Speicher-Transfer werden die Anschlüsse DB_{0-7} pro Byte viermal benötigt. Zunächst wird das zu lesende Adressen-High-Byte ausgegeben, anschließend fließt der Speicherinhalt in den 8237. Nachfolgend wird das zu schreibende Adressen-High-Byte ausgegeben, gefolgt vom zwischengespeicherten Wert,

			der nun in die neue Speicherstelle geschrieben wird.
$\overline{\text{IOR}}$	1	I/O	I/O Read. Die Funktion des Tri-State-Pins hängt davon ab, ob sich der Baustein im DMA-Modus befindet oder nicht. Im Nicht-DMA-Modus entspricht er dem $\overline{\text{RD}}$-Eingang eines Peripheriebausteins und wird von der CPU benutzt, um Daten aus den Registern zu lesen. Im DMA-Modus ist er Ausgang und wird benutzt, um Daten aus einem Peripheriebaustein zu lesen. Im kaskadierten DMA-Modus bleibt der Pin im inaktiven Zustand, im Tri-State.
$\overline{\text{IOW}}$	2	I/O	I/O Write. Die Funktion des Tri-State-Pins hängt davon ab, ob sich der Baustein im DMA-Modus befindet oder nicht. Im Nicht-DMA-Modus entspricht er dem $\overline{\text{WR}}$-Eingang eines Peripheriebausteins und wird von der CPU benutzt, um Daten in die Register zu schreiben. Im DMA-Modus ist er Ausgang und wird benutzt, um Daten in einen Peripheriebaustein zu schreiben. Im kaskadierten DMA-Modus bleibt der Pin im inaktiven Zustand, im Tri-State.
$\overline{\text{EOP}}$	36	I/O	End Of Process. Der $\overline{\text{EOP}}$-Pin besitzt intern einen Open-Drain-Transistor, so daß er sowohl als Ein- als auch als Ausgang benutzt werden kann. Er ist extern mit einem Pull-Up-Widerstand zu versehen. Wenn das Wort-Zählregister den Inhalt Null aufweist, wird am EOP-Pin ein kurzer Masse-Impuls ausgegeben (Ausnahme ist Kanal 0 bei einem Speicher-zu-Speicher-Transfer). Man nennt das einen internen End-Of-Process. Wird während eines DMA-Vorgangs der Pin von außen an Masse gelegt, führt das zu einem gewaltsamen Abbruch der DMA-Operation. Man nennt das einen externen End-Of-Process. Im kaskadierten Modus sind beide Fälle nicht möglich.

A0 A1 A2 A3	32 33 34 35	I/O I/O I/O I/O	Register Select Address. Diese vier Pins sind bidirektional. Im Nicht-DMA-Modus sind sie Tri-State-Eingänge und dienen der Auswahl der internen Register bei einem Zugriff der CPU auf den Baustein. Bei der Adressierung muß dem Rechnung getragen werden. Im DMA-Modus stellen sie lediglich die unteren vier Bits des Adressen-Low-Bytes dar. Im kaskadierten Modus bleiben sie im Tri-State.
A4-A7	37-40	O	Address. Diese Pins erwachen nur dann aus dem Tri-State zum Leben, wenn ein DMA aktiv ist, und führen dann die vier oberen Bits des Adressen-Low-Bytes. Sie geben zusammen mit den Pins A0-A3 beständig das LSB der Adresse aus und müssen somit nicht in einem Zwischenspeicher wie das Adressen-High-Byte aufgefangen werden.
HRQ	10	O	Hold Request. Bei einem DMA-Bedarf kann der DMA-Controller nicht augenblicklich mit der Datenübertragung beginnen, da die CPU erst ihren aktuellen Buszugriff ordnungsgemäß beenden muß. Das HRQ-Signal gibt der DMA-Controller an die CPU aus, um ihr mitzuteilen, daß der momentane Buszugriff ihr letzter ist und sie in den Halt-Zustand übergehen soll. Die Antwort erfolgt über den Eingang HLDA. Erst dann beginnt der Start des DMA-Zyklus. In Stand-Alone-Operationen, bei denen der 8237 beständig den Bus kontrolliert (Steuerung über CLK), ist der Ausgang HRQ mit dem Eingang HLDA zu verbinden, so daß die eine erforderliche Taktperiode zwischen beiden Signalen liegt.
DACK0 DACK1 DACK2 DACK3	25 24 17 16	O O O O	DMA Acknowledge. Mit einem aktiven Signal an diesen Ausgängen wird dem Peripheriebaustein angezeigt, daß seine DMA-Anforderung in Bearbeitung ist. Nach einem Reset sind die Ausgänge low-aktiv, können aber zu beliebiger Zeit durch Änderung des Inhalts des Befehlsregisters high-aktiv programmiert werden.

AEN	9	O	Address Enable. Das High-Byte der DMA-Adresse muß in einem Zwischenspeicher aufgefangen werden. Im Nicht-DMA-Modus sind seine Ausgänge im Tri-State, sonst mischen sie bei der Systemadressierung des Prozessors mit. Im DMA-Modus werden sie freigegeben, weil der 8237 nun die Buskontrolle besitzt. Diesem Zweck dient der Ausgang AEN, d.h. bei einem DMA-Transfer führt der Pin High- sonst Low-Pegel. Der Ausgang ist mit dem Enable-Eingang des Zwischenspeichers zu verbinden.
ADSTB	8	O	Address Strobe. Der AEN-Pin gibt zwar den Adreßzwischenspeicher frei, kann aber nicht bewirken, daß neue Werte in ihn geschrieben werden. Diese Aufgabe, den Adreßzwischenspeicher zu takten, übernimmt der Ausgang ADSTB. Unter zwei Umständen wird der Pin aktiv: 1. bei Beginn eines DMA-Transfers und 2. bei Änderung des Adressen-High-Bytes. Im Nicht-DMA-Modus oder im kaskadierten Modus bleibt der Ausgang an Masse.
$\overline{\text{MEMR}}$	3	O	Memory Read. Bei einem DMA-Transfer verläßt der Pin den Tri-State und gibt ein Signal aus, das zum Lesen des gewählten Speicherplatzes dient. Im kaskadiertenModus bleibt der Pin im Tri-State.
$\overline{\text{MEMW}}$	4	O	Memory Write. Bei einem DMA-Transfer verläßt der Pin den Tri-State und gibt ein Signal aus, das zum Beschreiben des gewählten Speicherplatzes dient. Im kaskadierten Modus bleibt der Pin im Tri-State.
NC	5	-	No Connect. Der Pin besitzt keine Funktion und sollte entweder mit +5 V verbunden sein oder offen gelassen werden.

Tabelle 4-9. Pin-Beschreibung des DMA-Controllers 8237

4.3.4 Die DMA-Operation

Ein Blick auf das Blockdiagramm (Bild 4-17) läßt in gewisser Weise die Funktion des DMA-Controllers erkennen. Die wichtigsten Komponenten sind die 16-Bit-Register und das temporäre 8-Bit-Register. Daneben sind weitere Register von Nöten, wie sie in allen programmierbaren Peripheriebausteinen auftreten. Es sind im wesentlichen Status-, Steuerungs-, Masken- und Anforderungsregister. Tabelle 4-10 stellt sie alle zusammen und gibt ihre Zahl und Bitbreite an. Die beiden letzten Spalten der Tabelle beziehen sich auf die Konversation mit der CPU. Die Funktion der internen Register ist später genauer beschrieben. Für das Verständnis der DMA-Transfertypen ist jedoch ein kurzer Überblick wichtig.

In den Basis-Registern stehen die von der CPU gewählten Voreinstellungen, die bei DMA-Start oder DMA-Ende in die aktuellen Register kopiert werden. Der Inhalt wird nur im temporären Byte-Zähler dekrementiert. Der temporäre 8-Bit-Speicher wird nur in einem Speicher-zu-Speicher-Transfer benutzt.

Name	Größe	Anzahl	lesbar	beschreibbar
Basis-Adreßregister	16	4	Nein	Ja
Basis-Byte-Zähler	16	4	Nein	Ja
Aktuelles Adreßregister	16	4	Ja	Ja
Aktueller Byte-Zähler	16	4	Ja	Ja
Statusregister	8	1	Ja	Nein
Befehlsregister	8	1	Nein	Ja
Temporärer Speicher	8	1	Ja	Nein
Modusregister	6	4	Nein	Ja
Maskenregister	4	1	Nein	Ja
Anforderungsregister	4	1	Nein	Ja

Tabelle 4-10. Die internen Register des 8237

Die Zeiteinheit der Operation des 8237 ist eine Taktperiode und wird im folgenden als State bezeichnet (Bild 4-18). Der DMA-Controller unterscheidet sieben verschiedene Arten von States.

Wenn der DMA-Controller für einen DMA-Transfer gesperrt ist oder keine Anforderung für einen freigegebenen Kanal vorliegt, befindet sich der DMA-Controller in Bereitscheft (Stand-By) und führt inaktive States S_i aus. Wird ein freigegebener DMA angemeldet, setzt der 8237 den Ausgang HRQ an Plus und wartet, bis ihm die Benutzung des Busses übertragen wird. In diesem Wartezustand führt der 8237 S_0-States aus. Sie werden so lange wiederholt, bis der Eingang HLDA an Plus geht. Gleichzeitig beginnt der DMA-Controller seine DMA-Operation. Zwischen der ausgegebenen Halteanforderung an den Prozessor HRQ und seiner Antwort HLDA muß mindestens eine Taktperiode liegen, d.h. zwei S_0-States müssen erfolgen.

In der Regel führt der 8237 bei einem DMA-Transfer pro übertragenes Byte drei oder vier States aus. Sie werden in dieser Reihenfolge mit S_1, S_2, S_3 und S_4 bezeichnet.

Im S_1-State wird der Adreßfreigabe-Pin AEN auf High-Pegel gesetzt, das Adressen-Low-Byte der Transferadresse an den Pins A_{0-7} ausgegeben und das Adressen-High-Byte an den Pins DB_{0-7}. Gleichzeitig erfolgt der Impuls zum Takten des Adressen-High-Bytes ADSTB. Der S_1-State wird nur unter drei Bedingungen ausgegeben:

1. bei Beginn eines jeden DMAs,
2. bei jedem Einzel-Byte-Transfer,
3. bei der Kopie eines Blocks nur dann, wenn sich das Adressen-High-Byte geändert hat.

Im S_2-State nun legt der 8237 entweder den Ausgang $\overline{MEMR}$ oder $\overline{IOR}$ an Masse. Wenn der S_1-State übersprungen wird, werden auch die Adreßausgänge A_{0-7} im S_2-State geändert.

Im S_3-State legt der 8237 entweder den Ausgang $\overline{MEMW}$ oder $\overline{IOW}$ an Masse.

Der S_4-State ist der letze State in einem Einzel-Byte-Transfer. Die Schreib-/Lesesteuersignale nehmen High-Pegel an und die Inhalte des aktuellen Adreßregisters und des aktuellen Byte-Zählers werden aktualisiert.

In einem kontinuierlichen DMA-Zyklus folgen nun die States S_1 oder S_2. Ist der DMA zu Ende, folgen S_i-States. Im Einzel-Byte-Modus folgt immer ein S_i-State nach dem S_4-State.

Mit dem Übergang zum S_i-State ändern gewisse Pins des 8237 ihren Zustand:

- Die Leitungen HRQ und AEN gehen an Masse.
- A_{4-7}, $\overline{DB_{0-7}}$, $\overline{MEMR}$ und $\overline{MEMW}$ gehen in den Tri-State.
- A_{0-3}, $\overline{IOR}$ und $\overline{IOW}$ werden als Eingänge konfiguriert.

Ist die Pulsbreite für gewisse Bausteine nicht ausreichend, kann sie durch Masse am Eingang Ready verbreitert werden. Während dieser Zeit führt der DMA-Controller Wait-States S_W aus.

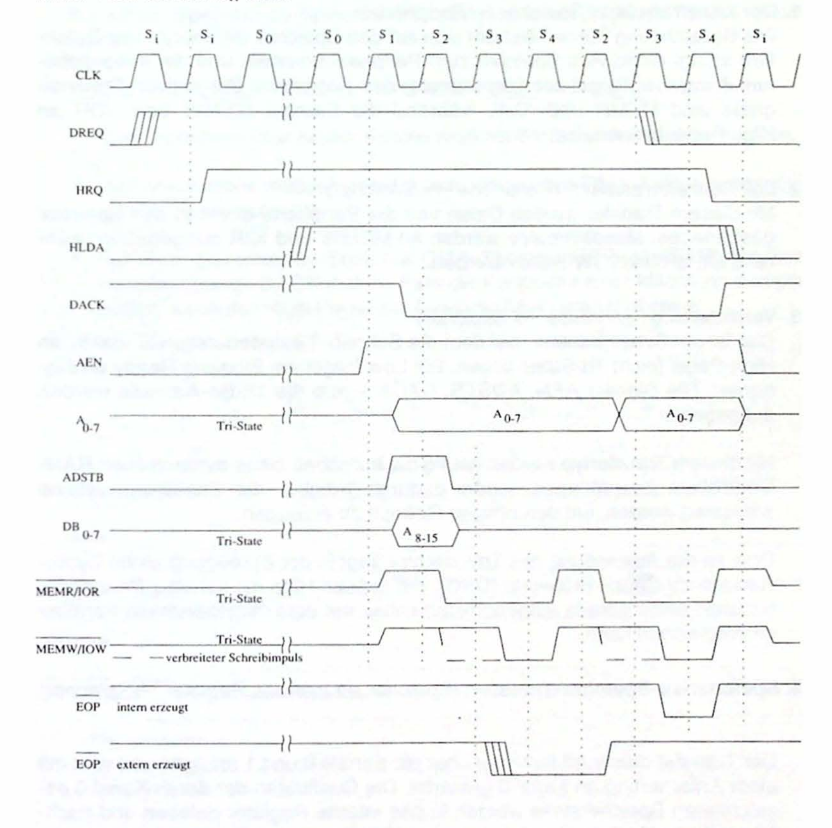


Bild 4-18. Zeitdiagramm für DMA-Transfers

4.3.5 Transfertypen

Der DMA-Controller 8237 kennt vier verschiedene Arten des DMA-Transfers:

1. **Der Lesetransfer:** (Speicher →Peripherie)
   Die Bezeichnung "Lese-" bezieht sich auf den Speicher (Memory). Der Datenfluß erfolgt direkt vom Speicher zum Peripheriebaustein und die ausgegebenen Adressen dienen zur Adressierung des Speichers. Die aktiven Steuersignale sind $\overline{MEMR}$ und $\overline{IOW}$, während die Signale $\overline{MEMW}$ bzw. $\overline{IOR}$ an High-Pegel bleiben.

2. **Der Schreibtransfer:** (Peripherie → Speicher)
   Mit diesem Transfer werden Daten von der Peripherie direkt in den Speicher geschrieben. Masseimpulse werden an $\overline{MEMW}$ und $\overline{IOR}$ ausgegeben, während $\overline{MEMR}$ und $\overline{IOW}$ inaktiv bleiben.

3. **Verifizierung:** (Adresse → Speicher)
   Das ist ein Scheintransfer, bei dem die Schreib-/Lesesteuersignale inaktiv an High-Pegel (nicht Tri-State) liegen. Ein Low-Pegel am Eingang Ready wird ignoriert. Die Signale AEN, ADSTB, DACK sowie die 16-Bit-Adresse werden ausgegeben.

   Mit diesem Transfertyp werden häufig die Aufgaben eines dynamischen RAM-Controllers übernommen, indem dadurch lediglich die Speicherbausteine adressiert werden, um den nötigen Refresh zu erzeugen.

   Eine zweite Anwendung des Transfertyps liegt in der Erzeugung eines Cyclic-Redunancy-Check-Prüfworts (CRC), mit dessen Hilfe ein serieller Peripheriebaustein seine gerade aufgenommen Daten mit dem mitgesendeten Paritätsbit vergleichen kann.

4. **Speicher-zu-Speicher-Transfer:** (Speicher → internes Register →Speicher)

   Der Transfer dieser Art kann nur über die Kanäle 0 und 1 erfolgen und wird mit einer Anforderung an Kanal 0 gestartet. Die Quelldaten der durch Kanal 0 bezeichneten Speicherstelle werden in das interne Register gelesen und nachfolgend von dort mit der Adresse von Kanal 1 in den Zielspeicherplatz ge-

schrieben. Dabei werden die Adressen von Kanal 0 und Kanal 1 alternierend ausgegeben, wobei nur die Signale $\overline{MEMR}$ gefolgt von $\overline{MEMW}$ aktiv sind. Dieser Transfer erfordert eine neue Kombination der States. Die Leseoperation besteht aus den States S_{11}, S_{12}, S_{13} und S_{14}, die Schreiboperation aus S_{21}, S_{22}, S_{23} und S_{24} (Bild 4-19).

Der eingestellte Transfertyp im Modusregister für Kanal 0 und 1 wird bei einem Speicher-zu-Speicher-Transfer ignoriert.

Bei einem Speicher-zu-Speicher-Transfer sind ferner folgende Punkte zu beachten:

- Die Inhalte der Byte-Zähler von Kanal 0 und 1 müssen identisch sein.
- Der Transfermodus beider Kanäle muß als Blocktransfer gesetzt werden.
- Alle Maskenbits müssen gesetzt sein, um externe DMA-Anforderungseingänge zu sperren.
- Bei einer Speicher-zu-Speicher-DMA-Operation sind alle DMA-Anerken nungsausgänge DACK inaktiv. Falls die Peripherie einer Mitteilung darüber bedarf, kann der High-Pegel am Ausgang AEN benutzt werden.

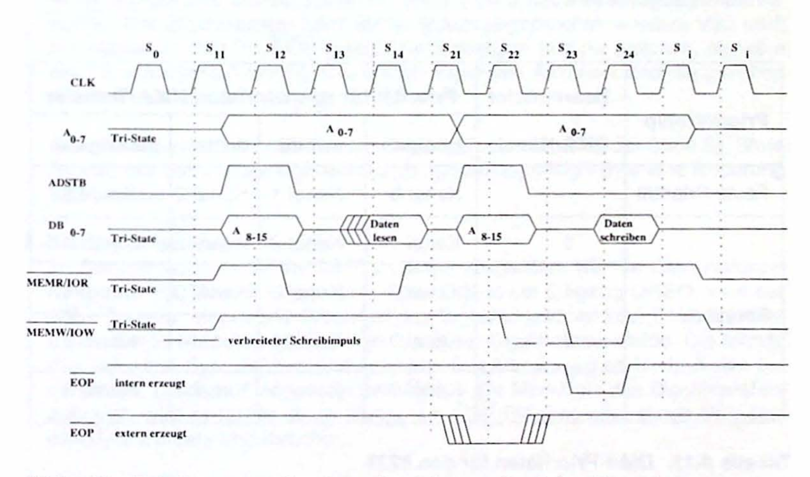


Bild 4-19. Zeitdiagramm für einen Speicher-zu-Speicher-Transfer

4.3.6 DMA-Prioritäten

Da der Fall gleichzeitig eintreffender DMA-Anforderungen nicht auszuschließen ist, muß der Baustein Vorkehrungen besitzen, die die Eingänge entsprechend ihrer Bedeutung auswählen. Treffen mehrere DMA-Anforderungen zur selben Zeit ein, wird ein DMA für den Kanal mit der höchsten Priorität ausgeführt (Tabelle 4-11). Allerdings kann ein laufender DMA-Zyklus nicht von einem höherer Priorität unterbrochen werden. Dieser wird erst nach Beendigung des vorausgehenden bedient. Es können im Befehlsregister Bit 4 zwei Arten der DMA-Priorität für den DMA-Controller 8237 programmiert werden:

1. Feste Priorität, Bit 4 = 0
   Kanal 0 hat immer die höchste Priorität, gefolgt von Kanal 1, Kanal 2 und Kanal 3 mit der niedrigsten Priorität.

2. Rotierende Priorität, Bit 4 = 1
   In diesem Prioritätsmodus wird dem Kanal, der zuletzt bedient wurde, die niedrigste Priorität zugewiesen. Die anderen Kanäle rücken in entsprechender Weise nach.

   Zum Beispiel: Nach der DMA-Bearbeitung von Kanal 1 bekommt Kanal 2 die höchste Priorität, gefolgt von Kanal 3 und Kanal 0; Kanal 1 wird die niedrigste Priorität zugewiesen.

Prioritätstyp	Bearbeiteter DMA-Kanal	Priorität für den nächsten DMA-Transfer			
		höchste	zweite	dritte	niedrigste
Feste Priorität	---	Kanal 0	Kanal 1	Kanal 2	Kanal 3
Rotation	0	Kanal 1	Kanal 2	Kanal 3	Kanal 0
	1	Kanal 2	Kanal 3	Kanal 0	Kanal 1
	2	Kanal 3	Kanal 0	Kanal 1	Kanal 2
	3	Kanal 0	Kanal 1	Kanal 2	Kanal 3

Tabelle 4-11. DMA-Prioritäten für den 8237

4.3.7 DMA-Transfermodi

Für alle Transfermodi muß das Anforderungssignal solange am DREQ-Eingang anliegen, bis der DMA-Controller mit seiner Arbeit begonnen hat, bis er also ein DACK-Signal ausgibt.

1. **Der Einzel-Byte-Transfer**:
   In diesem Modus überträgt der DMA-Controller nur ein einziges Byte und gibt nach Beendigung den Bus wieder frei. Der Byte-Zähler wird um eins vermindert und die Adresse je nach Programmierung um eins erhöht oder vermindert. Findet dabei ein Unterlauf (von 0 nach FFFF) im Byte-Zähler statt, wird ein Bit im Statusregister gesetzt und ein End-Of-Process-Impuls $\overline{EOP}$ erzeugt. Je nach Programmierung wird daraufhin entweder der ursprüngliche Zählwert aus dem Basis-Byte-Zählregister in den Byte-Zähler geschrieben oder das Maskierungsbit gesetzt, so daß ein Neustart jetzt mit dem Zählwert FFFF unmöglich wird. Bleibt das Anforderungssignal über diese Zeit hinaus aktiv, legt der 8237 mindestens einen inaktiven State S_i ein. Im allgemeinen wird die CPU diese Möglichkeit nutzen und vor der nächsten Busübergabe an den 8237 mindestens einen Maschinenbefehl ausführen.

2. **Der Blocktransfer**:
   Nach dem Start des DMA-Zyklus wird in diesem Modus der Datentransfer solange ausgeführt, bis ein Zählerunterlauf (von 0 nach FFFF) im Byte-Zähler auftritt. Der Blocktransfer kann ferner extern abgebrochen werden. Man muß nur Masse an den Pin $\overline{EOP}$ legen. Die Wirkungen sind die gleichen, als wäre ein Zählerunterlauf erfolgt. Die darauf folgenden Aktionen sind die gleichen wie im Einzel-Byte-Transfer.

   Bei kontinuierlichem DMA-Transfer folgt nach dem State S_4 der State S_2. State S_1 wird nur dann eingefügt, wenn sich im Adressen-High-Byte eine Änderung ergeben hat.

3. **Der Bedarfstransfer**:
   Im Bedarfsmodus wird der DMA so lange ausgeführt, wie die DMA-Anforderung aktiv ist. Verschwindet das aktive Signal am Eingang DREQ, wird der DMA-Transfer eingestellt. Erscheint das Signal erneut, wird der DMA-Transfer an der Stelle wieder aufgenommen, an der er abgebrochen wurde. Die Inhalte des aktuellen Byte-Zählers und des aktuellen Adreßregisters bleiben also unverändert. Da dieser Modus darüber hinaus alle Merkmale des Blocktransfers aufweist, wird er ferner durch Masse am $\overline{EOP}$-Eingang oder durch Unterlauf des Byte-Zählers abgebrochen.

4. Der kaskadierte Modus:

Dieser Modus wird benutzt, wenn die zur Verfügung stehenden vier Kanäle zur Versorgung des Systems nicht ausreichend sind. Es wird der Pin HRQ des Slaves mit dem Pin DREQ des Masters und der HLDA mit dem Pin DACK verbunden. Nur der Master wird in den kaskadierten Modus versetzt. Die Aufgabe des Masters besteht nur darin, die Busübergabe von der CPU zu fordern. Als Antwort er dies durch eine aktive DACK-Leitung dem Slave an.

Wenn die DMA-Anforderung (im Master) für den Kanal auftritt, der im kaskadierten Modus programmiert ist, und der Bus von der CPU übergeben wurde, wird lediglich die DACK-Leitung des Masters aktiv. Die Schreib-/Leseleitungen und die Adreßausgänge gehen in den Tri-State, AEN und ADSTB bleiben an Masse. Die Pegel der Eingänge $\overline{CS}$, $\overline{EOP}$ und READY werden ignoriert. Folgende Punkte sollten bei der Benutzung des Modus beachtet werden:

- Die Inhalte der Basis- und der aktuellen internen Register der Kanäle, die im kaskadierten Modus programmiert sind, nehmen zufällige Werte an.
- Eine DMA-Anforderung durch die Software für den Kanal, der irrtümlich für den kaskadierten Modus programmiert wurde, führt zu einer Verabschiedung des Systems.
- Zuerst sollte der Master initialisiert und der DACK-Ausgang auf high-aktiv gesetzt werden, da sonst die Register des Slaves nicht gelesen oder beschrieben werden können. Mit High-Pegel am Eingang HLDA wäre kein Zugriff auf den Slave-DMA-Controller mehr möglich.
- Mit einem externen $\overline{EOP}$ kann der Master nicht unterbrochen werden, aber sehr wohl der Slave.

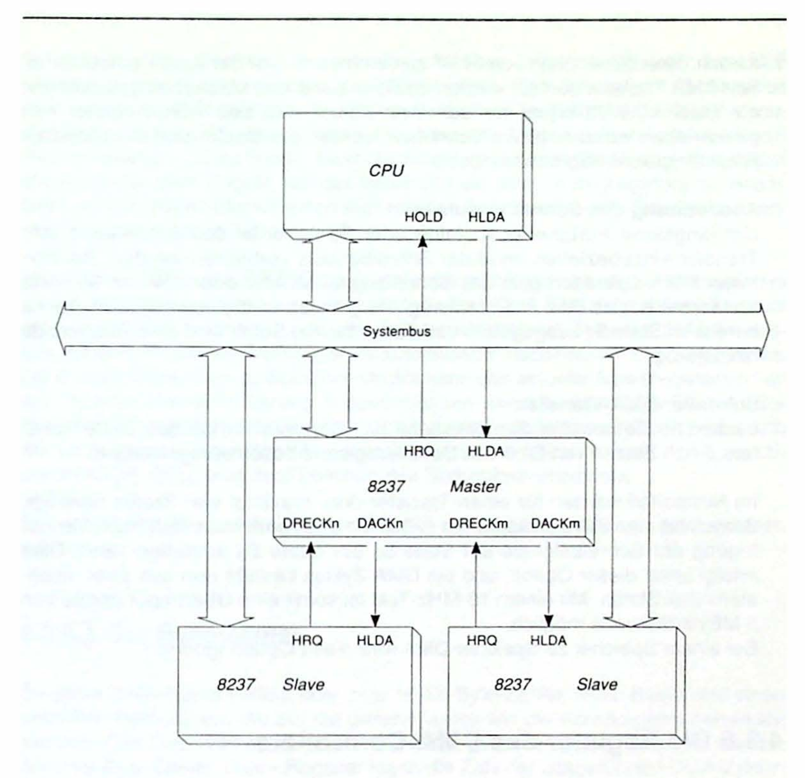


Bild 4-20. Beispiel eines kaskadierten DMA-Systems

Die nachstehenden Eigenschaften können auf die genannten Transfer-Modi programmiert werden:

- **Autoinitialisierung**:
  Die Autoinitialisierung kann mit Bit 4 des Modusregisters gewählt werden. Ist dieses Bit gesetzt, werden bei einem Unterlauf des aktuellen Byte-Zählers die unveränderten Inhalte des Basis-Byte-Zählregisters und des Basis-Adreßregisters in den aktuellen Byte-Zähler bzw. in das aktuelle Adreßregister kopiert. Das Maskenbit für diesen Kanal wird bei dieser Option nicht gesetzt, so daß nachfolgende DMA-Anforderungen sofort gestartet werden können. Wenn das Bit 4 gelöscht ist, wird bei einem Unterlauf des aktuellen Byte-Zählers das Maskenbit für den betreffenden Kanal gesetzt, um einen Neustart mit dem ak-

tuellen Byte-Zählerinhalt von FFFF zu verhindern. Soll der Kanal erneut für einen DMA-Transfer benutzt werden, müssen zuvor das Maskenbit gelöscht und die beiden 16-Bit-Werte für den Byte-Zähler und das Adreßregister neu geschrieben werden. (Beim Schreiben werden die Basis- und die aktuellen Register gleichzeitig beschrieben.)

- **Ausdehnung des Schreibimpulses:**
  Um langsame Peripheriebausteine oder Speicher in den schnellen DMA-Transfer einzubeziehen, muß der Schreibimpuls verbreitert werden. Bei normaler DMA-Operation geht das Schreibsignal $\overline{MEMW}$ oder $\overline{IOW}$ im S_3-State an Masse. Ist das Bit 5 im Befehlsregister gesetzt, wird diese negative Flanke bereits im State S_2 ausgegeben und dauert bis S_4. Somit wird eine Taktperiode hinzugefügt.

- **Schneller DMA-Transfer:**
  Lassen es die betreffenden Bausteine zu, kann das Tempo des DMA-Transfers durch Setzen von Bit 3 des Befehlsregisters beschleunigt werden.

  Im Normalfall werden für einen Transfer drei, maximal vier States benötigt. Betrachtet man das Zeitdiagramm in Bild 4-18, erkennt man, daß nach Vorverlegung der Schreibsignale auf State S_2 der State S_3 ausfallen kann. Dies erfolgt unter dieser Option, und ein DMA-Zyklus besteht nun aus zwei, höchstens drei States. Mit einem 10-MHz-Takt ist somit eine Übertragungsrate von 5 MByte/Sekunde möglich.
  Bei einem Speicher zu Speicher DMA wird diese Option ignoriert.

4.3.8 Die Register des DMA-Controllers

In der vorausgehenden Beschreibung der Arbeitsweise des DMA-Controllers wurde bereits häufig die Bedeutung der internen Register erwähnt. Das nun folgende ist eine Zusammenfassung der Register, ihrer Funktion und ihrer Verwendung.

4.3.8.1 Die Adreßregister

Jeder DMA-Kanal besitzt zwei 16-Bit-Register, die für die Adressierung verwendet werden, das Basis-Adreßregister und das aktuelle Adreßregister. Vor einer DMA-Operation müssen diese Register mit dem richtigen Wert beschrieben sein. Die Register sind nur dann beschreibbar, wenn der 8237 keinen DMA ausführt. Da der Baustein über die Eingänge DB_{0-7} nur über eine 8-Bit-Breite verfügt, muß der

Schreibvorgang zweimal an dieselbe Adresse erfolgen, zuerst das LSB (Adressen-Low-Byte), danach das MSB (Adressen-High-Byte). Beim Beschreiben hat man keine Wahl zwischen Basis- oder aktuellem Register; beide Register werden immer gleichzeitig beschrieben. Die Adressen für den entsprechenden Kanal sind in Tabelle 4-13 zu finden. Beim Auslesen der Adreßregister entzieht sich das Basisregister dem Zugriff; nur der Inhalt des aktuellen Adreßregisters ist lesbar. Die Leseadressen unterscheiden sich von den Schreibadressen. Eine Zusammenstellung findet sich in Tabelle 4-12.

Bei einem DMA wird zur Adressierung nur der Inhalt des aktuellen Adreßregisters an die Ausgänge geführt. Es wird während eines DMA-Transfers automatisch um eins erhöht oder vermindert und wird, falls gewünscht, am Ende des DMA-Prozesses mit dem Inhalt des Basisregisters automatisch wieder beschrieben. Nur im Kanal 0 und im Speicher-zu-Speicher-Modus kann das aktuelle Adreßregister von einer In- oder Dekrementierung ausgeschlossen werden, indem das Bit 1 im Befehlsregister gesetzt wird. Dadurch kann auf bequeme Weise ein Speicherbereich mit einer Konstanten aufgefüllt werden, z.B. der Textbildschirm mit dem Leerzeichen (ASCII=20_h), was dem Löschen des Bildschirms entspricht.

Der Inhalt des Basis-Adreßregisters bleibt unverändert und kann nicht gelesen werden.

4.3.8.2 Die Byte-Zähler

Ein jeder DMA-Kanal verfügt über zwei 16-Bit-Byte-Zähler, einen Basis- und einen aktuellen Byte-Zähler, die auf die gleiche Weise wie die Adreßregister behandelt werden. Das Beschreiben wirkt sich auf beide gleichzeitig aus, lesbar ist nur der aktuelle Byte-Zähler. Diese Register legen die Zahl der ausgeführten DMA-Zyklen fest. Zu beachten ist, daß die tatsächlich ausgeführte Zahl an Datenübertragungen um eins größer ist als der programmierte Wert, d.h. aus einem programmierten Wert von 100 resultieren 101 Übertragungen. Will man nur einen Transfer, sind die Byte-Zähler mit 0 zu beschreiben.

Bei einem DMA-Transfer wird der Inhalt des aktuellen Byte-Zählers um eins vermindert, eine Erhöhung des Wertes ist nicht möglich; auch wird sein Inhalt nicht nach außen geführt. Ein Unterlauf (von 0 nach FFFF) des Zählers bewirkt einen End-Of-Process-Impuls $\overline{EOP}$, der zur Autoinitialisierung der beiden aktuellen Register des betreffenden DMA-Kanals benutzt werden kann. Ist die Autoinitialisierung gesperrt, wird das Maskenbit für den betreffenden Kanal gesetzt. Für eine erneute Benutzung des Kanals muß das Maskenbit gelöscht und die Byte-Zähler neu beschrieben werden.

Der Inhalt des Basis-Byte-Zählers bleibt unverändert und kann nicht gelesen werden.

A_3 A_2 A_1 A_0	Adreß-FF	Kanal	Gelesenes Register
0 0 0 0	0	0	Bits 0-7 des aktuellen Adreßregisters
0 0 0 0	1	0	Bits 8-15 des aktuellen Adreßregisters
0 0 0 1	0	0	Bits 0-7 des aktuellen Byte-Zählers
0 0 0 1	1	0	Bits 8-15 des aktuellen Byte-Zählers
0 0 1 0	0	1	Bits 0-7 des aktuellen Adreßregisters
0 0 1 0	1	1	Bits 8-15 des aktuellen Adreßregisters
0 0 1 1	0	1	Bits 0-7 des aktuellen Byte-Zählers
0 0 1 1	1	1	Bits 8-15 des aktuellen Byte-Zählers
0 1 0 0	0	2	Bits 0-7 des aktuellen Adreßregisters
0 1 0 0	1	2	Bits 8-15 des aktuellen Adreßregisters
0 1 0 1	0	2	Bits 0-7 des aktuellen Byte-Zählers
0 1 0 1	1	2	Bits 8-15 des aktuellen Byte-Zählers
0 1 1 0	0	3	Bits 0-7 des aktuellen Adreßregisters
0 1 1 0	1	3	Bits 8-15 des aktuellen Adreßregisters
0 1 1 1	0	3	Bits 0-7 des aktuellen Byte-Zählers
0 1 1 1	1	3	Bits 8-15 des aktuellen Byte-Zählers
1 0 0 0	--	-	Statusregister
1 0 0 1	--	-	ungültig
1 0 1 0	--	-	ungültig
1 0 1 1	--	-	ungültig
1 1 0 0	--	-	ungültig
1 1 0 1	--	-	Temporäres Register
1 1 1 0	--	-	ungültig
1 1 1 1	--	-	ungültig

($\overline{CS}$=0, $\overline{RD}$=0, HLDA=0, $\overline{WR}$=1)

Tabelle 4-12. Gültige Adressen beim Lesen des 8237

A_3	A_2	A_1	A_0	Adreß-FF	Kanal	Beschriebenes Register
0	0	0	0	0	0	Bits 0-7 Basis- und aktuelles Adreßregister
0	0	0	0	1	0	Bits 8-15 Basis- und aktuelles Adreßregister
0	0	0	1	0	0	Bits 0-7 Basis- und aktueller Byte-Zähler
0	0	0	1	1	0	Bits 8-15 Basis- und aktueller Byte-Zähler
0	0	1	0	0	1	Bits 0-7 Basis- und aktuelles Adreßregister
0	0	1	0	1	1	Bits 8-15 Basis- und aktuelles Adreßregister
0	0	1	1	0	1	Bits 0-7 Basis- und aktueller Byte-Zähler
0	0	1	1	1	1	Bits 8-15 Basis- und aktueller Byte-Zähler
0	1	0	0	0	2	Bits 0-7 Basis- und aktuelles Adreßregister
0	1	0	0	1	2	Bits 8-15 Basis- und aktuelles Adreßregister
0	1	0	1	0	2	Bits 0-7 Basis- und aktueller Byte-Zähler
0	1	0	1	1	2	Bits 8-15 Basis- und aktueller Byte-Zähler
0	1	1	0	0	3	Bits 0-7 Basis- und aktuelles Adreßregister
0	1	1	0	1	3	Bits 8-15 Basis- und aktuelles Adreßregister
0	1	1	1	0	3	Bits 0-7 Basis- und aktueller Byte-Zähler
0	1	1	1	1	3	Bits 8-15 Basis- und aktueller Byte-Zähler
1	0	0	0	-	-	Befehlsregister
1	0	0	1	-	-	Software-DMA-Anforderungsregister
1	0	1	0	-	-	Schreiben eines einzelnen Maskenbits
1	0	1	1	-	-	Modusregister
1	1	0	0	-	-	Adreß-Flip-Flop löschen
1	1	0	1	-	-	Master Clear
1	1	1	0	-	-	Alle Maskenbits löschen
1	1	1	1	-	-	Alle Maskenbits beschreiben

($\overline{CS}$=0, $\overline{RD}$=1, HLDA=0, $\overline{WR}$=0)

Tabelle 4-13. Gültige Adressen beim Beschreiben des 8237

4.3.8.3 Das Modusregister

Jeder Kanal hat ein Modusregister, das 6 Bits umfaßt. Es kann nur unter einer Adresse 1011_b beschrieben werden. Wird der Modus für einen DMA-Kanal geschrieben, müssen dennoch alle 8 Bits berücksichtigt werden, denn die untersten zwei Bits, Bit 0 und Bit 1, geben die Nummer des Kanals an, dessen Modus durch die restlichen 6 Bits gewählt werden soll.

Die einzelnen Bits haben folgende Bedeutung:

Bit 7 Bit 6	Bit 7 und Bit 6 legen den DMA-Modus fest:	00 01 10 11	Bedarfstransfer Einzel-Byte-Transfer Blocktransfer Kaskadierter Modus
Bit 5	Verhalten des aktuellen Adreßregisters:	0 1	Abwärtszählen Aufwärtszählen
Bit 4	Autoinitialisierung:	0 1	Aus; Ein
Bit 3 Bit 2	Bit 3 und Bit 2 bestimmen den Transfertyp:	00 01 10 11	Verifizierung Schreibtransfer Lesetransfer unbenutzt
Bit 1 Bit 0	Bit 1 und Bit 0 wählen den Kanal aus:	00 01 10 11	Kanal 0 Kanal 1 Kanal 2 Kanal 3

Tabelle 4-14. Der Inhalt des Modusregisters

4.3.8.4 Das Befehlsregister

Es kann nur unter der Adresse 1000_b beschrieben, nicht aber gelesen werden. Nach einem Reset weist es den Inhalt 00_h auf.

Bit 7	Pegel für das DACK-Signal	0 High-aktiv 1 Low-aktiv
Bit 6	Pegel für das DREQ-Signal	0 High-aktiv 1 Low-aktiv
Bit 5	Verbreitertes Schreibsignal	0 Nein 1 Ja
Bit 4	Rotation der Priorität	0 Nein 1 Ja
Bit 3	Schneller Transfer	0 Nein 1 Ja
Bit 2	Freigabe des Bausteins	0 Nein 1 Ja
Bit 1	Keine Adreßänderung in Kanal 0	0 Nein 1 Ja
Bit 0	Speicher-zu-Speicher-Transfer	0 Nein 1 Ja

Tabelle 4-15. Der Inhalt des Befehlsregisters

4.3.8.5 Das Maskenregister

Das Maskenregister ist unter drei Adressen erreichbar. Unter der Adresse 1111_b hat man gleichzeitig Zugang zu allen Maskenbits (Tabelle 4-16), mit der Adresse 1010_b kann man jedes Maskenbit einzeln setzen oder löschen (Tabelle 4-17). Jedes gesetzte Bit schaltet den entsprechenden Kanal für einen DMA-Transfer aus. Nach dem Schreiben eines beliebigen Wertes an die Adresse 1110_b sind alle vier Maskenbits rückgesetzt und jeder Kanal für einen DMA freigegeben. Bei einem Reset oder Master-Clear werden alle Bits gesetzt und verhindern jegliche DMA-Aktion. Befindet sich ein Kanal nicht im Autoinitialisierungsmodus, bewirkt ferner der Unterlauf des aktuellen Byte-Zählers oder ein externes $\overline{EOP}$-Signal das Setzen des betreffenden Maskenbits.

Die Bits 7 bis 4 haben keine Bedeutung.			
Bit 3	Maskenbit für Kanal 3	0 1	Kanal frei Kanal gesperrt
Bit 2	Maskenbit für Kanal 2	0 1	Kanal frei Kanal gesperrt
Bit 1	Maskenbit für Kanal 1	0 1	Kanal frei Kanal gesperrt
Bit 0	Maskenbit für Kanal 0	0 1	Kanal frei Kanal gesperrt

Tabelle 4-16. Der Inhalt des Maskenregisters unter der Adresse 1111_b

Die Bits 7 bis 3 haben keine Bedeutung.		
Bit 2	Maskenbit den für in Bit 0 und Bit 1 genannten Kanal	0 Kanal frei 1 Kanal gesperrt
Bit 1 Bit 0	Auswahl des Kanals	00 Kanal 0 01 Kanal 1 10 Kanal 2 11 Kanal 3

Tabelle 4-17. Der Inhalt des Maskenregisters unter der Adresse 1010_b

4.3.8.6 Das Anforderungsregister

Das ist ein 4-Bit-Register, das unter der Adresse 1001_b beschrieben, aber nicht gelesen werden kann. Mit ihm kann jeweils nur ein Bit gesetzt oder gelöscht werden (Tabelle 4-18).

Der DMA-Controller kann auf zwei Arten von DMA-Anforderungen reagieren:

1. Hardware-Anforderung. Sie wird durch ein aktives Signal am Eingang DREQ ausgelöst und kann durch das Maskenbit gesperrt werden.
2. Software-Anforderung. Das geschieht mit dem Setzen des Anforderungsbits durch die CPU. Sie kann nicht durch ein Maskenbit verhindert werden.

Wie die Hardware-Anforderung unterliegt auch die Software-Anforderung der Prioritätsauswahl durch eine interne Logik. Sie kann eine Block- oder Einzel-Byte-Übertragung starten. Zum Start eines Speicher-zu-Speicher-DMAs ist das Anforderungsbit von Kanal 0 zu setzen. Nach dem Setzen des Anforderungsbits sollte man nicht versäumen, alle Maskenbits zu setzen, um externe Anforderungen, die eventuell eine höhere Priorität genießen, auszuschließen.

Nach der DMA-Operation eines Kanals oder nach einem Reset werden alle Anforderungsbits gelöscht.

Die Bits 7 bis 3 haben keine Bedeutung.		
Bit 2	Software-DMA-Anforderung gesetzt	0 Nein 1 Ja
Bit 1 Bit 0	Auswahl des Kanals	00 Kanal 0 01 Kanal 1 10 Kanal 2 11 Kanal 3

Tabelle 4-18. Der Inhalt des Anforderungsregisters

4.3.8.7 Das Statusregister

Das Statusregister umfaßt 8 Bits und kann unter der Adresse 1000_b nur gelesen werden. Die oberen vier Bits spiegeln unabhängig vom Maskenbit den Zustand der Eingänge DREQ wieder. Eine Eins bedeutet, am DREQ-Eingang liegt ein aktives Signal an. Durch Setzen aller Maskenbits und nach Auswertung dieser Information kann die Software durch gezieltes Löschen eines Maskenbits die Priorität selbst setzen.

Die unteren vier Bits zeigen mit einer Eins an, daß für den entsprechenden Kanal ein Byte-Zähler-Unterlauf stattfand oder ein externes $\overline{\text{EOP}}$-Signal anlag. Beim Lesen, nach einem Reset oder Master-Clear werden die unteren 4 Bits gelöscht.

4.3.8.8 Das temporäre Register

Es ist ein 8-Bit-Register, das unter der Adresse 1101 nur gelesen werden kann. Ein versehentliches Schreiben an diese Adresse bewirkt einen Master-Clear. Es wird als Byte-Zwischenspeicher bei einem Speicher-zu-Speicher-DMA-Transfer benutzt. Beim Lesen des Registers erfährt die CPU den Inhalt des letzen übertragenen Bytes, falls es nicht durch einen Reset oder Master-Clear gelöscht wurde.

Bit 7	DREQ-Eingang für Kanal 3 ist aktiv.	0 Nein 1 Ja
Bit 6	DREQ-Eingang für Kanal 2 ist aktiv.	0 Nein 1 Ja
Bit 5	DREQ-Eingang für Kanal 1 ist aktiv.	0 Nein 1 Ja
Bit 4	DREQ-Eingang für Kanal 0 ist aktiv.	0 Nein 1 Ja
Bit 3	Ein Zählerunterlauf ist in Kanal 3 erfolgt.	0 Nein 1 Ja
Bit 2	Ein Zählerunterlauf ist in Kanal 2 erfolgt.	0 Nein 1 Ja
Bit 1	Ein Zählerunterlauf ist in Kanal 1 erfolgt.	0 Nein 1 Ja
Bit 0	Ein Zählerunterlauf ist in Kanal 0 erfolgt.	0 Nein 1 Ja

Tabelle 4-19. Der Inhalt des Statusregisters

4.3.9 Software-Befehle

Der DMA-Controller 8237 kennt drei Software-Befehle, die dadurch ausgeführt werden, daß die entsprechende Adresse angelegt und ein Schreibimpuls ausgeführt wird. Der Inhalt des dabei auf dem Datenbus anstehenden Bytes ist belanglos, da er nicht in das Innere des Bausteins vordringt. Entscheidend für den Befehl ist die richtige Adressierung.

Die Befehle im einzelnen:

- **Adreß-Flip-Flop löschen**
  Der Zustand des Flip-Flops steuert beim Schreiben oder Lesen der 16-Bit-Register den 8-Bit-Datenfluß. Ist es gelöscht, erfolgt der Zugriff auf das LSB, ist es gesetzt, erfolgt der Zugriff auf das MSB. Nachdem das LSB geschrieben oder gelesen wurde, wird das Flip-Flop automatisch gesetzt, so daß der nächste Zugriff auf das High-Byte erfolgt. In diesem Zustand bleibt es aber, so daß es für einen neuen Zugriff auf das LSB eines 16-Bit-Registers per Befehl gelöscht werden muß. Es wird durch einen Schreibbefehl an die Adresse 1100_b gelöscht.

- **Master-Clear**
  Dieser Befehl mit der Adresse 1101_b hat die gleiche Wirkung wie ein Reset:
  - Alle Maskenbits für die DMA-Kanäle werden gesetzt.
  - Das Befehlsregister wird mit Nullen beschrieben (Bit 2!).
  - Das temporäre Register wird gelöscht.
  - Die vier niedrigsten Bits des Statusregisters werden gelöscht.
  - Das Adreß-Flip-Flop wird gelöscht.
  - Das Software-Anforderungsregister wird gelöscht.

- **Maskenregister löschen**
  Dieser Befehl an die Adresse 1110_b löscht alle Bits des Maskenregisters und gibt jeden Kanal für eine DMA-Übertragung frei.

4.3.10 Anwendungsbeispiele

Das erste Beispiel zeigt eine der üblichen Methoden für die Konfigurierung eines DMA-Systems mit dem DMA-Controller 8237 und einem 8080/85-Mikroprozessorsystem, das mit kleinen Änderungen auch zur Unterstützung der Mikrocontroller geeignet ist (Bild 4-21). Der DMA-Controller gibt bei Erscheinen eines internen oder externen DMA-Wunsches die Halteanforderung HRQ an den Prozessor, die sobald als möglich vom Prozessor über die HLDA-Leitung quittiert wird. Daraufhin übernimmt der DMA-Controller den Adreß-, Daten- und Kontrollbus. Diese Leitungen können direkt miteinander verbunden sein, da entweder die entsprechenden Ausgänge des Prozessors oder die des DMA-Controllers im Tri-State sind und sich somit nicht gegenseitig stören.

Die Adresse für den ersten Datentransfer erscheint am 8237 mit zwei Bytes. Das LSB erscheint an den Ausgängen A_{0-7}, das MSB auf dem Datenbus DB_{0-7}. Der Inhalt des Datenbusses wird mittels ADSTB-Signal in einen Adreßzwischenspei-

cher getaktet, um die 16 Bits des Adreßbusses zu vervollständigen. Der Inhalt des Adreßzwischenspeichers wird nur dann aktualisiert, wenn entweder ein Über- oder Unterlauf im LSB eintritt.

Das zweite Beispiel ist in Bild 4-22 gezeigt. Der DMA-Controller befindet sich in einem 8088-Mikroprozessor-System. Der Systemtakt wird durch den Taktgenerator 8284 erzeugt, dem Prozessor und nach Invertierung dem DMA-Controller zugeführt. Die Pegelinvertierung muß vorgenommen werden, um das Taktsignal den erforderlichen High- und Low-Zeiten des 8237 anzupassen. Mit den vier OR-Gates unterhalb des Prozessors werden die für den Speicher und I/O-Bereich erforderlichen Kontrollsignale erzeugt. Die Auswahl des DMA-Controllers und des Speichers erfolgt durch den Dekoder 82C339, der in einer für das System geeigneten Weise programmiert ist. Da das MSB der Adresse über den Datenbus sowohl vom Prozessor als auch vom DMA-Controller ausgegeben wird, sind für das Multiplexen der Adresse zwei Zwischenspeicher 8282 (oder 74HC373) vorgesehen. Das Quittierungssignal des Prozessors HLDA und die Adreßspeicherfreigabe durch den DMA-Controller sind geodert auf den Freigabe-Pin des linken Bausteins geführt, um zu gewährleisten, daß es während des DMA-Betriebs zu keinen Buskonflikten kommt.

Wird nun von dem Peripheriebaustein ein DMA-Bedarf angemeldet und hat die CPU dem DMA-Controller die Busübernahme gestattet, wird der Peripheriebaustein als Antwort über die Leitung DACK selektiert. Die Steuersignale gibt nun der DMA-Controller aus, die über die vier OR-Gatter mit negativer Logik (= NAND-Gatter) den direkten Datenfluß zwischen Speicher und Peripherie steuern. Die Daten fließen nicht durch den DMA-Controller. Da sowohl die CPU im Halt-Status als auch der DMA-Controller im inaktiven Zustand die Steuersignale im Tri-State lassen, müssen noch an jede Kontrolleitung Pull-Ups hinzugefügt werden, die im Bild 4-22 nicht gezeigt sind.

Der DMA-Controller 8237 ist Bestandteil der Personal-Computer mit MS-DOS-Betriebssystem und dort an den I/O-Adressen 0 bis 0F anzusprechen. Die Hauptaufgabe neben dem Datenfluß von oder zur Floppy-Disk oder Festplatte besteht mangels DRAM-Controller in der Erzeugung der für die dynamischen RAMs erforderlichen Refresh-Zyklen. Zu diesem Zweck löst der Timer-Baustein 8253 alle 2 ms an Kanal 0, der die höchste Priorität genießt, eine DMA-Anforderung aus. Der DMA-Controller adressiert nun die dynamischem RAMs in einem Block von 0 bis FF_h (für den Refresh ausreichend). Da für einen Refresh keine Daten fließen, sondern lediglich die Speicher adressiert werden müssen, genügt als Transfertyp die Verifizierung.

Da der DMA-Controller 8237 nur einen Bereich von 64 KByte selbstständig adressieren kann, muß die CPU vor einem DMA nicht nur das Adreßregister und den Byte-Zähler beschreiben, sondern auch das Page- oder Seitenregister, für das der 4-zu-4-Register-File-Baustein 74LS670 verwendet wird.

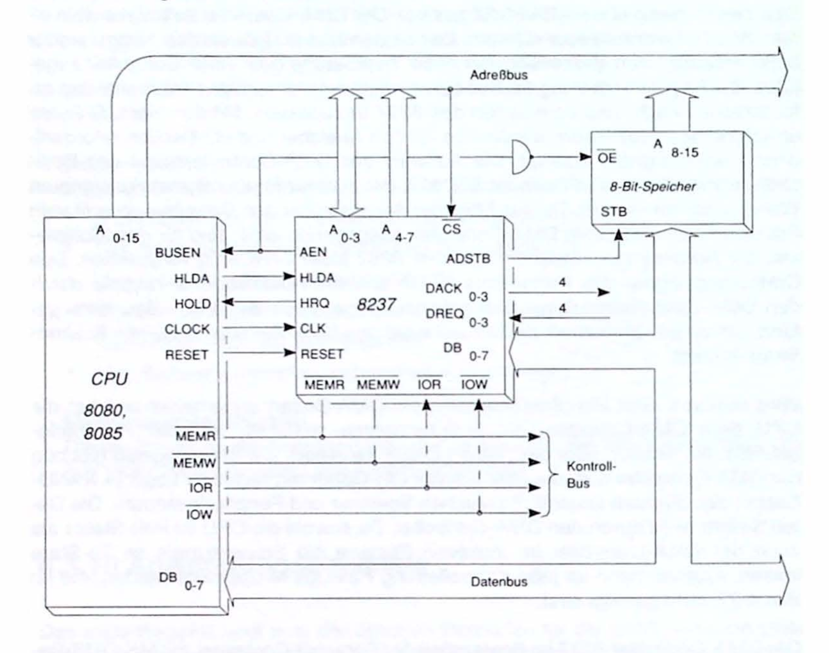


Bild 4-21. 8237-System-Interface mit dem 8080/8085-Prozessor

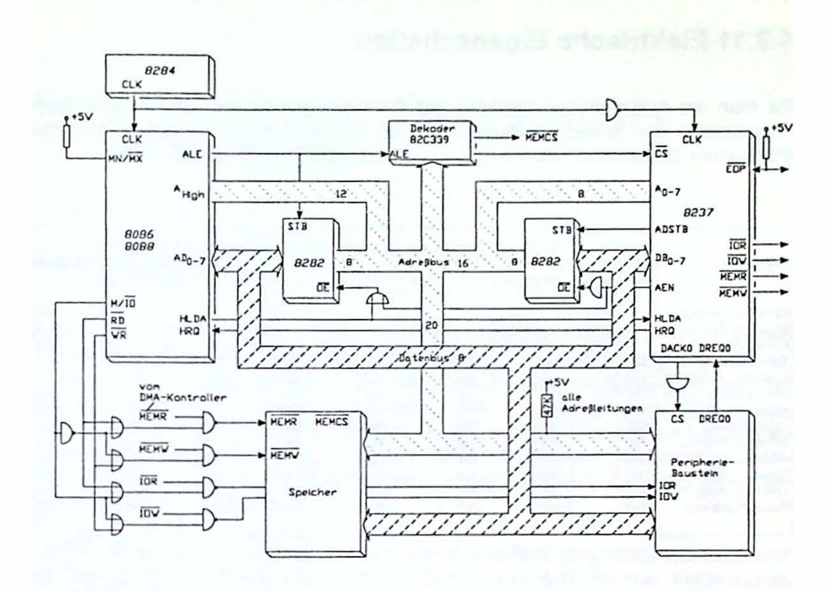


Bild 4-22. Hardware-Beispiel für einen Speicher-I/O-DMA

4.3.11 Elektrische Eigenschaften

Da man als Anwender ein berechtigtes Interesse daran hat, ob die einzelnen Impulszeiten den restlichen Bausteinen des Systems gerecht werden, sind die wichtigsten Zeitdaten in Tabelle 4-20 zusammengefaßt.

Parameter	8237		8238A-4		8237A-5		82C37AP-4		82C37AP-5		82C37A-5		82C37A		Einheit
	min.	max.	min.	max.	min.	max.	min.	max.	min.	max.	min.	max.	min.	max.	
Stromversorgung		130		130		130		15		15		15		15	mA
Stand-By-Strom		110		110		110		0,01		0,01		0,01		0,01	mA
Frequenz		3,12		4		5		4		5		4		8	MHz
Taktperiode	320		250		200		250		200		200		125		ns
CLK-High-Pulsbreite	120		100		80		100		80		70		55		ns
CLK-Low-Pulsbreite	150		110		68		110		68		50		43		ns
Adressen vor $\overline{RD}$	50		50		50		50		50		10		10		ns
$\overline{RD}$-Pulsbreite	300		250		200		250		200		200		155		ns
$\overline{CS}$ vor Schreiben	200		150		130		150		130		0		0		ns
Adressen vor $\overline{WR}$	200		150		130		150		130		0		0		ns
Daten vor $\overline{WR}$	200		150		130		150		100		150		100		ns
$\overline{WR}$-Pulsbreite	200		200		160		200		160		150		100		ns
Reset-Pulsbreite	300		300		300		300		300		300		300		ns

Tabelle 4-20. Die wichtigsten elektrischen Eigenschaften der 8237er-Bausteine

4.4 Der Controller für dynamische RAMs 82C08

4.4.1 Übersicht

Dynamische RAMs sind für eine zuverlässige Aufbewahrung der in ihnen gespeicherten Daten auf eine mehr oder weniger periodische Auffrischung angewiesen, im folgenden **Refresh** genannt. Die Beschreibung der Vorgänge im Innern der dynamischen RAMs findet sich im Kapitel 1.4.4. Ein solcher Vorgang muß üblicherweise innerhalb von 2 ms erfolgen, da sonst Speicherinhalte verlorengehen können. Des weiteren muß die Adresse in Reihen und Spalten unterteilt und nachfolgend in das dynamische RAM getaktet werden. Diese Arbeit übernimmt ein dynamischer RAM-Controller.

Der 82C08 ist auf die verschiedensten Systemanforderungen programmierbar und bei verschiedenen Frequenzen für die Zusammenarbeit mit den Prozessoren 86/88, 186/188 und 286 ohne Wait-State optimiert. Er versorgt 64 KByte und 256 KByte dynamische RAMs mit einer Organisation von 256K x 1 bzw. 256K x 4 und ist zu dynamischen RAMs mit statischen Spalten oder Ripple-Modus kompatibel; er unterstützt nicht den schnellen Transfer-Modus dieser RAMs. Ferner ist er in der Lage, ohne zusätzliche Treiberbausteine einen Bereich von 1 MByte direkt zu adressieren und den nötigen Strom für den Refresh zu liefern, den er in fünf verschieden Modi ausführen kann. Selbst im Power-Down-Modus kann er mit Hilfe eines Batteriestroms die Refresh-Zyklen aufrechterhalten und verfügt über einen automatischen RAM-Warm-Up. Unter einem Warm-Up versteht man acht Refresh-Zyklen nach einem Reset zur Initialisierung der RAMs. Der 82C08 ist die CMOS-Version des 8208 und mit ihm pinkompatibel. Drei Pins (Nrn. 17, 22 und 48) sind unterschiedlich und unterstützen in der CMOS-Version die Leistungsreduzierung. Sie versorgen hauptsächlich den Power-Down-Modus, in dem der Baustein nur geringen Strom zieht. Mit einem eigenen Refresh-Takt an Pin 22 kann man die Vorteile der RAMs nutzen, die einen flexibleren Refresh erlauben. Ferner gibt es gegenüber dem 8208 geringe Änderungen in der Zeittaktsteuerung. Um die externen Busspeicher überflüssig zu machen, wurden die WE- und $\overline{\text{CAS}}$-Signale gekürzt. Diese zeitlichen Unterschiede sind jedoch abwärts-kompatibel zu Systemen mit dem 8208.

Wegen der besonderen Eigenschaft, auch im Power-Down die Speicher mit dem nötigen Refresh zu versorgen, wird an dieser Stelle die CMOS-Version des DRAM-Controllers beschrieben.

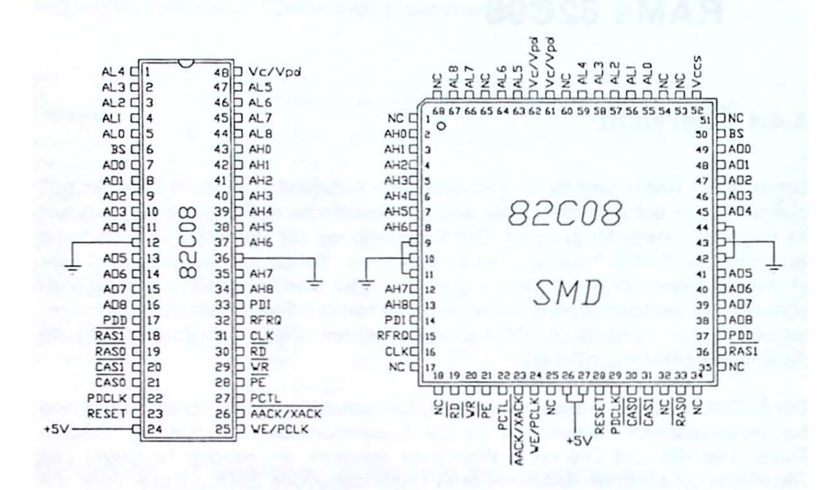


Bild 4-23. Pin-Belegung des 82C08

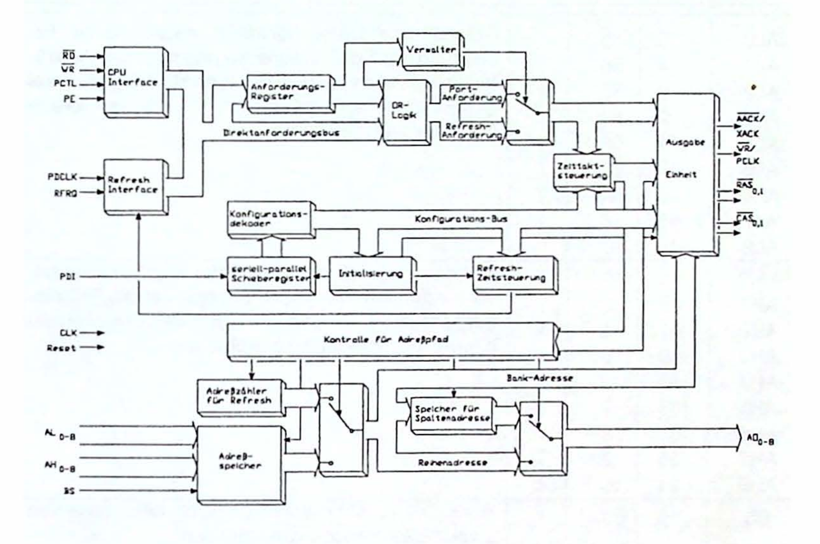


Bild 4-24. Blockdiagramm des 82C08

4.4.2 Die Funktionen der Pins

Symbol	Pin-Nummer		Art	Name und Funktion
	DIP	SMD		
AL0 AL1 AL2 AL3 AL4 AL5 AL6 AL7 AL8	5 4 3 2 1 47 46 45 44	55 56 57 58 59 63 64 66 67	I I I I I I I I I	Adresse Low. Diese Adreßbits werden für die Erzeugung der Spaltenadresse des internen Adreßmultiplexers genutzt. Im 286er Modus, d.h. bei gesetztem Programmdatenbit 0, werden diese Bits intern gespeichert.
AH0 AH1 AH2 AH3 AH4 AH5 AH6 AH7 AH8	43 42 41 40 39 38 37 35 34	2 3 4 5 6 7 8 12 13	I I I I I I I I I	Adresse High. Diese Adreßbits werden benutzt, um für den internen Adreßmultiplexer die Reihenadresse zu erzeugen. Im 286er Modus werden diese Eingänge intern gespeichert.
BS	6	50	I	Bank Select. Mit diesem Eingang kann eine von zwei Bänken ausgewählt werden.
AO0 AO1 AO2 AO3 AO4 AO5 AO6 AO7 AO8	7 8 9 10 11 13 14 15 16	49 48 47 46 45 41 40 39 38	O O O O O O O O O	Adreßausgänge. Über diese Ausgänge werden die Reihen- und Spaltenadressen, die entweder von der CPU oder dem internen Refresh-Zähler stammen, an die dynamischen RAMs weitergegeben. Sie können die RAM-Bausteine direkt ohne zusätzlich Stromverstärker treiben. Man schaltet lediglich einen Serienwiderstand in jede Adreßleitung, um die Impedanzen anzupassen.

$\overline{\text{RAS0}}$ $\overline{\text{RAS1}}$	19 18	33 36	O O	Reihenadresse Strobe. Mit Hilfe dieser Signale speichern die dynamischen RAMs die Reihenadresse, die an den Pins AO0 bis AO8 ausgegeben wird. Welches $\overline{\text{RAS}}$-Signal aktiv ist, entsheidet der Zustand des Bank-Select-Eingangs
$\overline{\text{CAS0}}$ $\overline{\text{CAS1}}$	21 20	30 31	O O	Spaltenadresse Strobe. Mit Hilfe dieser Signale speichern die dynamischen RAMs die Spaltenadresse, die an den Pins AO0 bis AO8 ausgegeben wird. Welcher Ausgang aktiv ist, entscheidet der Zustand des Bank-Select-Eingangs
RESET	23	28	I	Reset. Durch High-Pegel an diesem Eingang werden alle internen Zähler auf Null gestellt. Unmittelbar nach dem Reset wird das Programmdatenwort seriell über den PDI-Pin eingelesen. Gleichzeitig werden die logischen Zustände der PCTL- und RFRQ-Pins gespeichert und programmieren so den 82C08. Danach wird der acht Zyklen dauernde Warm-Up der dynamischen RAMs ausgeführt.
WE/ PCLK	25	24	O	Write Enable/Programm Takt. Unmittelbar nach dem Reset gibt dieser Pin neun Taktsignale aus, um das neun Bit umfassende Programmdatenwort über den PDI-Pin einzulesen. Nach dieser Programmierung ist der Pin **high**-aktiv und dient zur Schreibfreigabe der RAMs.
$\overline{\text{AACK}}$/ $\overline{\text{XACK}}$	26	23	O	Advance Acknowledge/Transfer Acknowledge. Wenn das Programmdatenbit 8 eine Null enthält, zeigt dieser Ausgang dem Prozessor an, daß er seine Arbeit fortsetzen und auf Daten zugreifen kann. Diese Signalform läßt sich mit Hilfe des Programmbits 1 für ein bestimmtes System optimieren. Im allgemeinen wird dieser Ausgang mit dem Ready-Eingang des Systems verbunden und zeigt an, ob der Prozessor mit seinen Aktivitäten fortfahren kann. Setzt man das Programmbit 8, entspricht die Zeittaktsteuerung an diesem Ausgang dem Multibus-kompatiblen $\overline{\text{XACK}}$-Signal.

PCTL	27	21	I	Port Control. Der logische Pegel an diesem Pin wird mit der fallenden Flanke des Reset-Signals gespeichert und dient somit der Programmierung des Bausteins. Mit ihm trifft der Baustein die Auswahl zwischen Status (S_{0-2}) oder Befehlssignalen. Führt der Pin bei Reset High-Pegel, werden die Eingänge $\overline{WR}$, $\overline{RD}$ und PCTL mit den Statusleitungen S_{0-2} des Prozessors verbunden. Die Signale werden intern dekodiert. Mit Masse bei Reset werden die $\overline{WR}$- bzw. die $\overline{DR}$-Leitung des Systems mit den gleichnamigen Eingängen des 82C08 verbunden. PCTL bleibt dabei an Masse.
$\overline{PE}$	28	21	I	Port Enable. Mit Masse an diesem Eingang wird der DRAM-Controller für eine Speicherzugriff freigegeben. In seiner Funktion entspricht er den Chip-Select-Eingängen $\overline{CS}$ anderer Bausteine und wird normalerweise mit einem Adreßdekoder verbunden.
$\overline{WR}$	29	20	I	Write. Dieser Eingang wird entweder mit der S_0- oder der $\overline{WR}$-Leitung verbunden.
$\overline{RD}$	30	19	I	Read. Dieser Eingang wird entweder mit der S_1- oder der $\overline{RD}$-Leitung verbunden.
CLK	31	16	I	Clock. Takteingang, der entweder vom Systemtakt oder einem separaten Takt gespeist wird.
RFRQ	32	15	I	Refresh Request. Der Pegel an diesem Eingang wird bei Reset gespeichert und dient damit der Programmierung des Bausteins. Mit High-Pegel ist der Baustein für einen internen Refresh oder externen Refresh mit Ausfallschutz vorbereitet. Masse schaltet den internen Refresh-Zeitgeber, ab und es kann nur noch ein externer Refresh ohne Ausfallschutz ausgelöst werden. Ein konzentrierter oder Blockrefresh ist ebenfalls nicht mehr möglich. Der logische Pegel wird ebenfalls abgefragt, wenn der Baustein in den Power-Down-Modus übergeht.

PDI	33	14	I	Programm Data Input. Über diesen Eingang werden die Bits des Programmworts bei Reset in den Baustein geschrieben. Für Standardeinstellungen kann er entweder mit Masse oder +5 V verbunden sein. Für eine individuelle Programmierung verwendet man ein externes Schieberegister.
*PDD	17	37	I	Power Down Detect. High-Pegel versetzt den Baustein in den Power-Down- Modus, Low-Pegel versetzt ihn in den aktiven Modus.
*PDCLK	22	29	I	Power Down Clock. Im Power-Down-Modus sorgt der Takt an diesem Eingang für einen Refresh der dynamischen RAMs.
*V_{PD}	48	61, 62	I	Voltage Power Down. Batteriegespeiste Spannungsversorgung für den Power-Down-Modus. Im aktiven Zustand mit +5 V verbunden.
+5V	24	26, 27	I	Stromversorgung für den aktiven Zustand. Im Power- Down-Modus inaktiv.
Masse: 12,36/9,10,11,42,43,44.				
NC nur SMD: 1,17,18,25,32,34,35,51,53,54,60,65,68				
V_{CCS} nur SMD: 52				

* Unterschiede zur Nicht-CMOS-Version.
Der 8208 erfordert an den Pins 17, 22 Masse und an Pin 48 +5 V.

Tabelle 4-21. Die Beschreibung der Pins des 82C08

4.4.3 Die Zusammenarbeit mit dynamischen RAMs

Die RAM-Adressierung:

Die Hauptaufgabe des DRAM-Controllers ist die Unterstützung des Prozessors durch Übernahme der Aufgaben, die dynamische RAMs an ein Computersystem stellen. Das sind die gemultiplexte Adressierung und der Refresh. Der 82C08 liest die von der CPU ausgegeben Adresse und spaltet sie in die Reihen- und Spaltenadresse auf. Beide werden nacheinander über die Leitungen A0-8 an die RAMs ausgegeben. Dabei bilden die neun obersten Bits AH0-8 die Reihenadresse und die neun unteren Adreßbits AL0-8 die Spaltenadresse. Um Echos oder Rauschen auf den Leitungen zu vermeiden, müssen serielle Widerstände in die Leitungen eingefügt werden. Der DRAM-Controller ist für eine Zusammenarbeit mit 16-K-, 64-K- und 256-K-DRAMs geeignet. Bild 4-25 zeigt die Verbindung des Adreßbusses zum 82C08 unter Benutzung verschiedener DRAM-Größen. Dabei sollten nichtausgewiesene Adreßeingänge mit High-Pegel verbunden sein. Die Adreßleitung A0 dient zusammen mit dem High-Byte-Freigabesignal $\overline{BHE}$ der Auswahl eines Bytes im Prozessorwort (Bild 4-32 und Bild 4-33).

Dadurch, daß die Adreßleitung A_1 mit dem Bankauswahleingang BS verbunden ist, wird mit jedem nachfolgenden Speicherzugriff die Bank gewechselt, wodurch sich aus der Sicht des Prozessors die Zugriffszeiten wesentlich verringern und sich der Datenfluß beschleunigen läßt. Langsame und preiswertere Bausteine können durch diesen Trick scheinbar kürzere Zugriffszeiten entwickeln als schnelle. Für jede Bank gibt der 82C08 ein eigenes $\overline{RAS}$- und $\overline{CAS}$-Signal aus. Diese Organisation gestattet die eben geschilderte Zeitüberlappung bei einem RAM-Zugriff. Während noch die adressierten RAM-Bausteine der einen Bank die Daten lesen oder ausgeben, beginnt der DRAM-Controller bereits mit der Vorbereitung des Speichers in der anderen Bank.

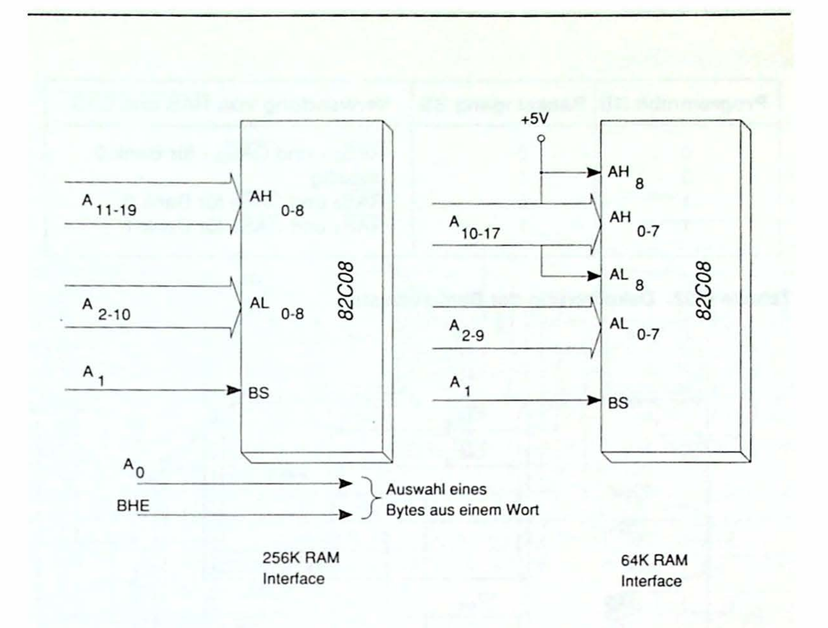


Bild 4-25. Die Verwendung der Adreßleitungen durch den DRAM-Controller

Wie in Bild 4-26 gezeigt, kann der 82C08 zwei getrennte Speicherbänke versorgen. Wenn nur eine Bank vorhanden ist, kann er so umprogrammiert werden, daß er in nur einer Bank einen größeren Adreßbereich anspricht (Tabelle 4-22), oder er kann in der einen Bank eine Bitbreite von 32 Bits versorgen (Bild 4-27). Die beiden $\overline{RAS}$- und $\overline{CAS}$-Signale erscheinen dann zu derselben Zeit. Das Programmbit RB wird nicht benutzt, um den Bankauswahleingang BS zu überprüfen. Die Hardware muß so beschaffen sein, daß ein Zugriff auf eine nicht existierende Bank verhindert wird. Das kann man mit einer externen Logik erreichen, die den Portfreigabe-Pin PE desaktiviert, wenn auf eine illegale Bank zugegriffen werden soll.

Programmbit RB	Bankeingang BS	Verwendung von $\overline{RAS}$ und $\overline{CAS}$
0	0	$\overline{RAS_{0,1}}$ und $\overline{CAS_{0,1}}$ für Bank 0
0	1	ungültig
1	0	$\overline{RAS_0}$ und $\overline{CAS_0}$ für Bank 0
1	1	$\overline{RAS_1}$ und $\overline{CAS_1}$ für Bank 1

Tabelle 4-22. Dekodierung der Bankauswahl

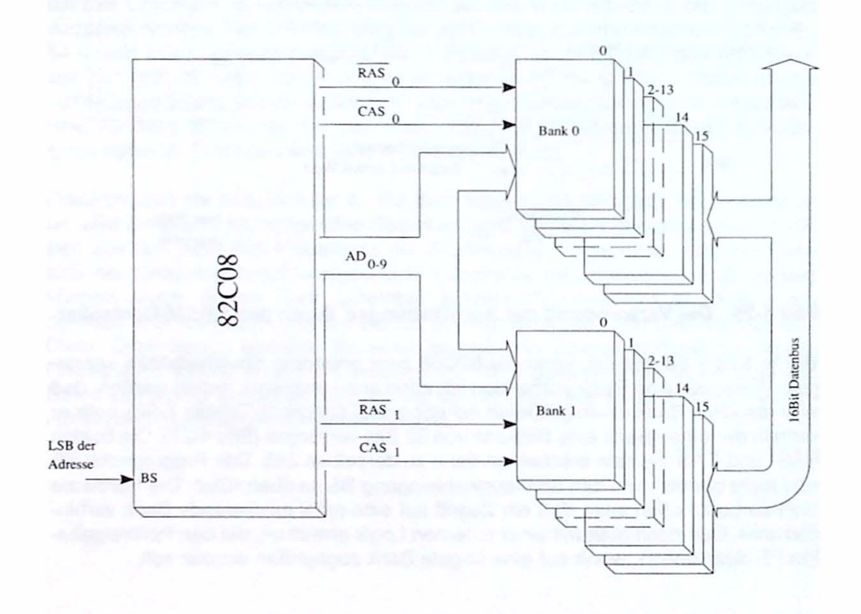


Bild 4-26. Der 82C08 mit einer Zwei-Bank-Konfiguration

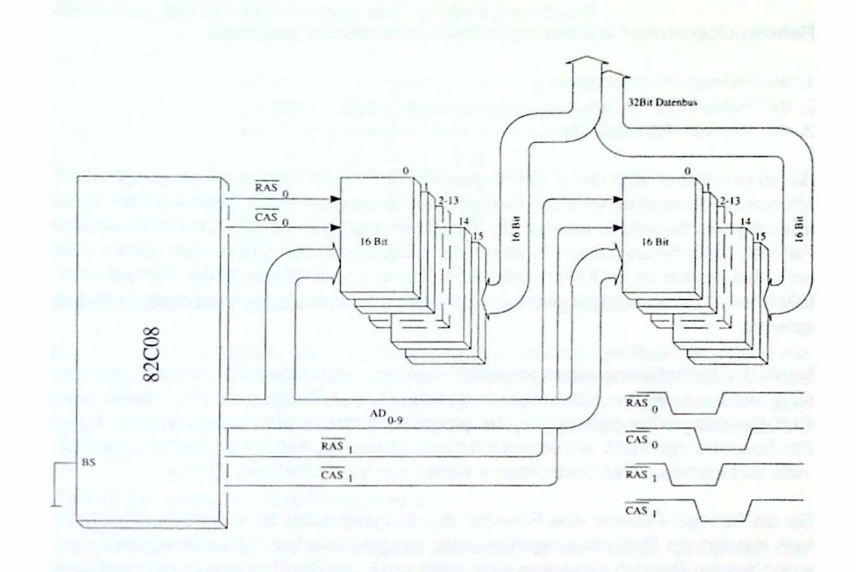


Bild 4-27. Der 82C08 mit einem einzigen 32-Bit-Speicher

4.4.4 Der Refresh

Refresh-Operationen werden durch drei interne Blöcke gesteuert:

1. der Refresh-Kontrollblock
2. der Refresh-Zeitgeber
3. der Refresh-Adreßzähler

Bei einem Reset wird der Zustand des RFRQ-Pins im Kontrollblock gespeichert. Mit dieser Information wird der Refresh-Modus programmiert. Während der Initialisierung des Bausteins erzeugt der Zeitgeber nach jeweils 32 CLK-Perioden eine Refresh-Anforderung. Danach legen die programmierten Werte das Zeitintervall fest, das jedoch 32 CLK-Perioden nicht überschreitet. Wenn keine Refresh-Ausfallsicherung programmiert wurde, wird das durch den Zeitgeber erzeugte Signal ignoriert.

Nach der Feststellung einer entweder internen oder externen Refresh-Anforderung wird ein Bit im Kontrollblock gesetzt. Die Anforderung wird daher eine CLK-Periode später bearbeitet. Bei programmierter Ausfallsicherung wird dabei der Zeitgeber gelöscht, um erneut mit dem Zählen des minimalen Sicherheitsintervalls zu beginnen. Der Kontrollblock startet nun einen Refresh-Zyklus.

Da die Refresh-Periode eine Funktion des Eingangstakts ist, ist die Geschwindigkeit, mit der ein Refresh ausgeführt wird, programmierbar. Diese Frequenz kann einen weiten Bereich abdecken und somit muß der DRAM-Controller in entsprechender Weise die Refresh-Periode anpassen. Mit den Bits CFS und FFS des Programmworts wird der Vorteiler des Refresh-Zählers automatisch auf den richtigen Wert eingestellt (Tabelle 4-23).

CFS	FFS	Taktfrequenz	Prozessor
0	0	5 MHz	86,88,186,188
0	1	8 MHz	86,88,186,188
1	0	10 MHz	286
1	1	16 MHz	286

Tabelle 4-23. Die Programmierung des Refresh-Zählers

Für Frequenzen, die dazwischen liegen, ist mit den Programmbits "Zählintervalle" CI0 und CI1 eine Feinabstimmung des Refresh-Intervalls möglich. Damit ist eine Verkürzung der Refresh-Periode laut Tabelle 4-24 möglich.

CI0	CI1	Refresh-Verkürzung
0	0	0 %
0	1	10 %
1	0	20 %
1	1	30 %

Tabelle 4-24. Die Feinabstimmung des Refresh-Intervalls

Intern besitzt der 82C08 einen 9-Bit-Adreßzähler, der für den Refresh verwendet wird. Er kann Standard-DRAMs versorgen, die einem Refresh von entweder 128 Reihen in 2 ms, 256 Reihen in 4 ms oder 512 Reihen in 8 ms erfordern. Außerdem kann mit dem Programmbit 6 PLS ein Refresh von 256 Reihen in 2 ms gewählt werden.

Der 82C08 unterstützt fünf Refresh-Arten.

Der **verteilte Refresh** ist verbreitetste Methode. Er wird durch Verbinden des Refresh-Anforderungs-pins RFRQ mit High-Pegel gewählt. In diesem Modus führt der DRAM-Controller automatisch die Refresh-Zyklen aus, ohne daß der Prozessor oder eine externe Logik dazu eingreifen müßte. Die Refresh-Rate ist programmierbar und wird durch die Bits 0, 4, 5, 6 und 7 des Programmworts festgelegt.

Beim **versteckten Refresh** besteht die Gefahr, daß Daten verloren gehen, wenn der Prozessor über längere Zeit keinen Zugriff auf den Speicher nimmt. Um dem vorzubeugen, besitzt der 82C08 eine Ausfallsicherung, die, falls sie programmiert ist, nach Ablauf des internen Zählers von selbst einen Refresh-Zyklus auslöst. Der versteckte Refresh findet seine Verwendung bei langsam arbeitenden Prozessoren, denn die modernen Hochleistungsprozessoren stellen nicht mehr die für einen Refresh notwendige Zeit nach dem Einlesen des Operationskodes zur Verfügung.

Um den DRAM-Controller in diese Betriebsart zu versetzen, muß der Refresh-Anforderungs-Pin RFRQ unmittelbar nach dem Reset High-Pegel führen. Über eine externe Logik müssen die Statusleitungen des Prozessors dekodiert werden und dem Refresh-Anforderungs-Pin RFQ ein Signal in Form einer positiven Flanke zugeführt werden.

Die dritte Refresh-Art ist der **externe Refresh ohne Ausfallsicherung**. Er erlaubt dem Anwender nach eigenen Wünschen einen Refresh auszulösen. Um in diesen Modus zu gelangen, muß der Refresh-Anforderungs-Pin RFRQ unmittelbar nach dem Reset Low-Pegel führen. Dadurch wird der interne Zeitgeber desaktiviert. Um nun einen einzelnen Refresh zu erzeugen, muß an dem Eingang RFRQ eine positive Flanke erfolgen, wobei der High-Pegel mindestens eine Taktperiode lang stabil anliegen muß. Liegt er allerdings mehr als nur eine Taktperiode lang an, wird nicht nur ein einzelner Refresh an einer Adresse, sondern ein konzentrierter Refresh des gesamten Speichers ausgelöst. Verhindern läßt sich dieser ungewollte Effekt, wenn das anfordernde Signal extern mit dem Taktsignal synchronisiert wird. Versäumt es die externe Logik, rechtzeitig einen Refresh auszulösen, besteht in diesem Falle allerdings kein Schutz vor Datenverlust.

Die vierte Refresh-Möglichkeit ist die Erzeugung eines konzentrierten Refreshs. Wie aus dem vorangehenden Absatz hervorgeht, muß nur der Eingang RFRQ bei Reset Masse und später mindestens zwei Taktperioden lang High-Pegel führen, damit ein **konzentrierter Refresh** erzeugt wird. Dabei führt der DRAM-Controller ein Block von 128 aufeinanderfolgenden Refresh-Zyklen aus. Der interne 9-Bit-Refresh-Adreßzähler wird nach jedem Einzel-Refresh um eins erhöht. Der konzentrierte Refresh wird beendet, wenn der Refresh-Adreßzähler den Wert XX111 1111 (X=0 oder 1) erreicht. Wenn der Adreßzähler vor dem Block-Refresh, z.B. durch Einzel-Refresh, den Inhalt XX111 1101 aufweist, werden nur drei Refresh-Zyklen ausgeführt.

Die letzte Refresh-Möglichkeit ist trivial und heißt **kein Refresh**. Ein dynamischer Speicher muß nicht ausdrücklich mit einem Refresh versorgt werden, wenn gewährleistet ist, daß innerhalb von 2 ms jede Reihe des DRAMs adressiert wird. Fälle dieser Art treten auf, wenn ein Grafikprozessor den Inhalt des Pixelspeichers ständig zur Anzeige bringt. Diesen Modus erreicht man, indem man den Eingang RFRQ fest mit Masse verbindet.

Treten eine Speicheranforderung und ein Refresh zusammen auf, so hat der Refresh Vorrang und der Speicherzugriff wird nicht bedient, solange der Refresh ausgeführt wird. Das wird dem Prozessor mittels $\overline{\text{AACK}}$-Signal angezeigt.

4.4.5 Die Programmierung

Die Programmierung der DRAM-Controllers erfolgt hardware-mäßig während eines Resets und ist nachfolgend durch den Prozessor nicht mehr änderbar. Dabei werden die Zustände dreier Pins (PCTL, RFRQ und PDI) intern gespeichert. Der DRAM-Controller benötigt für eine optimale Arbeitsweise zusätzliche Informationen bezüglich des Interface-Typs (synchron oder asynchron), Taktrate, Geschwindigkeit der dynamischen RAMs, etc. Besondere Aufmerksamkeit gilt dabei dem Programm-Data-Input-Pin PDI, da über ihn die 9 Bits des Programmworts eingelesen werden. Der DRAM-Controller ist so konstruiert, daß mit Masse oder +5 V am PDI-Pin Grundeinstellungen für gewisse Prozessorentypen programmiert werden können. Dadurch reduziert sich die Hardware-Anforderung zur Programmierung wesentlich.

Durch Masse am PDI-Pin werden alle Bits des Programmworts mit Nullen beschrieben. Diese Grundeinstellung konfiguriert den 82C08 für die Zusammenarbeit mit einem 8-MHz-Prozessor (8086, 8088, 80186, 80188) und 150-ns-DRAMs oder einem 10-MHz-Prozessor mit 120-ns-DRAMs ohne Wait-States. Die Refresh-Rate beträgt im letzteren Fall 11,8 µs. Mit High-Pegel am PDI-Pin werden alle Bits des Programmworts mit Einsen beschrieben und es ist eine maximale Frequenz von 20 MHz möglich. In diesem Modus ist er für eine Zusammenarbeit mit dem 80286 bei 8 MHz und 120-ns-DRAMs oder 10 MHz und 100-ns-DRAMs ohne Wait-States geeignet. Wenn der 20-MHz-82C08 im Standardmodus programmiert ist, beträgt die Refresh-Periode 11,8 µs.

Im langsamen Modus, erstes Programmbit ist Null, arbeitet der DRAM-Controller mit derselben Frequenz wie der Prozessor (8-MHz-82C08 für einen 8-MHz-Prozessor). Wenn der Baustein im schnellen Modus programmiert ist, erstes Programmbit ist eine Eins, arbeitet der 82C08 mit der doppelten Frequenz wie der Prozessor (16-MHz-82C08 für einen 8-MHz-80286).

Mit den restlichen Bits des Programmworts kann eine Feinabstimmung auf die Gegebenheiten des Systems vorgenommen werden (Tabelle 4-25). Allerdings ist dazu etwas Hardware nötig. Ein Beispiel wird in Bild 4-28 gezeigt. Der Reset-Impuls dient zum parallelen Laden des Schieberegisters. Unmittelbar nach einem Reset gibt der Ausgang WE/PCLK neun Taktsignale aus, die vorgesehen sind, um aus einem externen Schieberegister (74HC165) die Programmbits seriell über den Eingang PDI in den Baustein zu takten. Nur während dieser Zeit nimmt der Baustein die Programmbits auf. Dabei wird das erste serielle Datenbit in das Bit 0 des Programmworts geschrieben, das neunte in Bit 8. Nachfolgende Schreibsignale, die ebenfalls über den Pin WE/PCLK ausgegeben werden, führen zu keiner Veränderung des Programmworts. Die Programmierung des 82C08 mittels Schieberegister muß erfolgt sein, bevor der Rest des Systems aus dem Reset-zustand

entlassen wird. Das ist nicht nötig, wenn eine der Standardprogrammierungen mit Plus oder Masse verwendet wird. Eine Schaltung, die den DRAM-Controller früher aus dem Reset-Zustand entläßt, zeigt Bild 4-29.

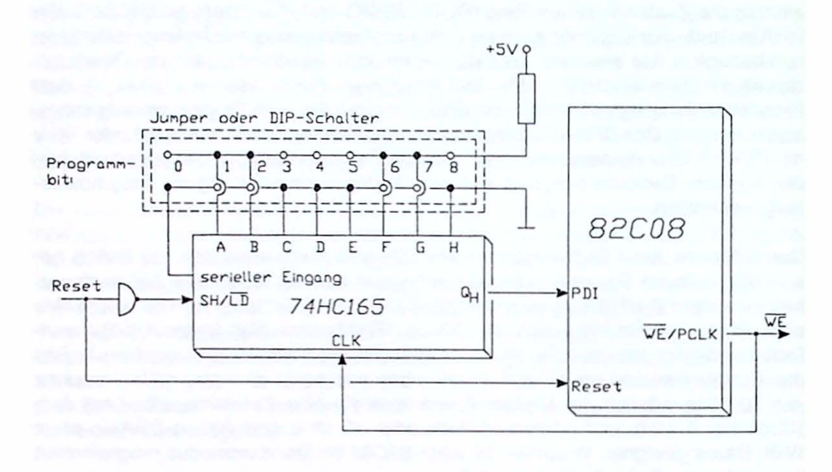


Bild 4-28. Die Programmierung des 82C08 mittels externem Schieberegister

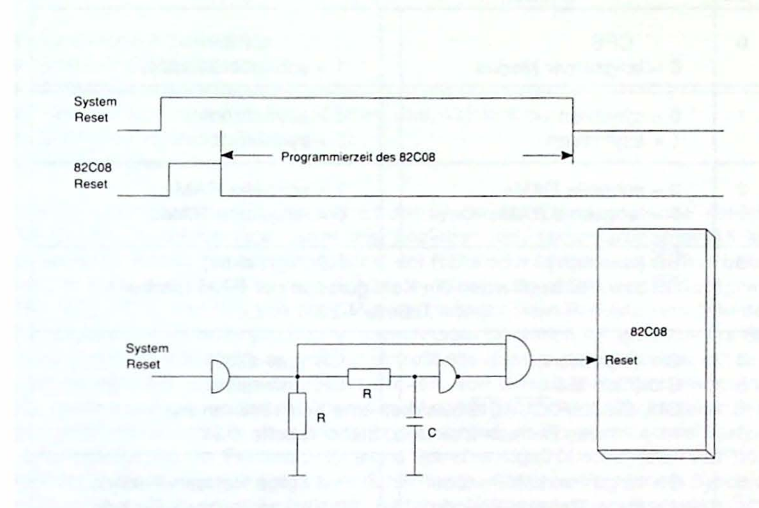


Bild 4-29. Verkürzung des Reset-Impulses

Bit	Funktion	
0	CFS 0 = langsamer Modus:	CFS 1 = schneller Modus:
1	0 = synchron 1 = asynchron	1 = asynchron 0 = synchron
2	0 = schnelle RAMs 1 = langsame RAMs	1 = schnelle RAMs 0 = langsame RAMs
3	$\overline{RB}$ (low-aktiv) $\overline{RB}$ bzw. RB bestimmen die Konfiguration der RAM-Bänke. Siehe Tabelle 4-22	RB (high-aktiv)
4 5	CI1 (high-aktiv) CI0 (high-aktiv) CI1, CI0 bzw. $\overline{CI1}$, $\overline{CI0}$ bewirken eine Feinabstimmung des Refresh-Intervalls. Siehe Tabelle 4-24	$\overline{CI1}$ (low-aktiv) $\overline{CI0}$ (low-aktiv)
6	0 = lange Refresh-Periode 1 = kurze Refresh-Periode	1 = lange Refresh-Periode 0 = kurze Refresh-Periode
7	$\overline{FFS}$ 0 = schnelle CPU-Frequenz 1 = langsame CPU-Frequenz Siehe Tabelle 4-23	FFS 1 = schnelle CPU-Frequenz 0 = langsame CPU-Frequenz
8	Funktion von Pin 26: 0 = $\overline{AACK}$ 1 = $\overline{XACK}$	1 = $\overline{AACK}$ 0 = $\overline{XACK}$

Tabelle 4-25. Das Programmwort des 82C08

Der Inhalt der Programmworts ist so beschaffen, daß beim Beschreiben mit Nullen oder nur Einsen Standardkonfigurationen hergestellt werden. In beiden Fällen sind folgende Standardbedingungen programmiert:

- Synchrone Arbeitsweise
- Schnelle RAM-Speicher
- 2 RAM-Bänke vorhanden
- 128 Reihen-Refresh in 2 ms; 256 in 4 ms, 512 in 8 ms
- Schnelle Prozessor-Taktfrequenz
- $\overline{AACK}$-Signal

Eine programmierbare Eigenschaft ist der synchrone oder asynchrone Betrieb. Die Begriffe synchron bzw. asynchron beziehen sich vereinbarungsgemäß auf dynamische RAMs, die sich entweder in der Nähe oder fern des Controllers befinden. Im Falle des 82C08 bezieht sich die synchrone Operation auf Steuersignale ($\overline{RD}$, $\overline{WE}$, PCTL und $\overline{PE}$) von der CPU, die in einer festen Relation zum Takt des Prozessors stehen, während bei der asynchronen Operation keine besondere Beziehung zum Takt besteht. Asynchron eintreffende Signale müssen zuerst mit dem eigenen internen Takt synchronisiert werden. Von diesem Punkt an werden die asynchrone und die synchrone Operation identisch gehandhabt. Die kürzeste Synchronisationszeit umfaßt eine Taktperiode, während die langsamste zwei Taktperioden beträgt. Da der Prozessor für einen Speicherzugriff üblicherweise vier Taktperioden oder weniger benötigt, wird durch den zusätzlichen Takt für die Synchronisation die Zeit um 25 Prozent gedehnt (vorausgesetzt, Prozessor und DRAM-Controller arbeiten mit demselben Takt). Somit sind synchrone Controller immer dann vorzuziehen, wenn die Umgebung es gestattet.

4.4.6 Die Speicherinitialisierung

Sobald der DRAM-Controller den Ladevorgang des Programmworts beendet hat, beginnt er mit der Speicherinitialisierung, indem er acht Speicheraufwärmezyklen oder Warm-Ups ausführt:

1. Unmittelbar dem Reset folgend zeigt der Initialisierungsblock (Bild 4-24) den restlichen internen Schaltkreisen an, daß der DRAM-Controller in der Initialisierungsperiode ist.
2. Mit diesem Signal wird der Refresh-Zeitgeber gestartet. Solange der Initialisierungsblock das Signal aktiv hält, veranlaßt der Refresh-Zeitgeber nach jeweils 32 Taktperioden einen Einzel-Refresh. Diese Intervallgröße wurde deswegen gewählt, um die Stromaufnahme bei der Initialisierung möglichst gering zu halten.
3. Bei Reset wird der Adreßzähler, der in einem Refresh-Zyklus die Reihenadresse erzeugt, auf den Wert $FFF8_h$ eingestellt und sein Inhalt mit der dritten Taktperiode eines jeden Refresh-Zyklus um eins erhöht. Erreicht sein Inhalt den Wert $FFFF_h$, gibt er ein Signal an den Initialisierungsblock, um das Ende der Initialisierungsperiode anzuzeigen. Somit werden acht Speicheraufwärmzyklen ausgeführt.

Während der Programmierung und der Speicheraufwärmung bei Reset nimmt der 82C08 keine externen Anforderungen entgegen. Somit muß der Prozessor von Beginn der Programmierung des DRAM-Controllers bis zu seinem ersten Speicherzugriff eine Zeit von 37 µs (f = 8 MHz) warten. Das sollte bei der Berechnung des RC-Gliedes nach Bild 4-29 berücksichtigt werden. Eine Reset-Verkürzung ist also nicht nur dann empfehlenswert, wenn der DRAM-Controller mittels Schieberegister programmiert wird, sondern auch um ihm Zeit zur DRAM-Initialisierung zu geben.

4.4.7 Die Zusammenarbeit mit dem Prozessor

Intern besitzt der 82C08 Schaltkreise, die in der Lage sind, mit Kontrollsystemen verschiedener Prozessoren zusammenzuarbeiten. Wie erwähnt kann der DRAM-Controller synchron oder asynchron zum Prozessortakt arbeiten. Befindet er sich in einem System mit den Prozessoren 8086/88/186/188/286, ist die synchrone Arbeitsweise die optimale. Im asynchronen Modus führt der 82C08 die notwendige Synchronisation für die $\overline{RD}$- und $\overline{WR}$-Eingänge aus. Über die Programmierung durch den Port-Controll-Eingang PCTL ist er in der Lage, die Statusleitung S_{0-2} der Prozessoren direkt zu dekodieren. Darüber hinaus kann er so programmiert werden, daß er Multibus-Signale oder Signale von einem Buscontroller akzeptiert. Dabei muß er nicht notwendigerweise mit derselben Taktfrequenz wie der Prozessor betrieben werden.

Wird bei Reset am PCTL-Eingang Masse festgestellt, ist der 82C08 nachfolgend für alle Operationen im Befehlsmodus, d.h. der $\overline{WR}$- und der $\overline{RD}$-Pin wird mit der gleichnamigen Steuerleitung des Prozessors oder des Buscontrollers verbunden. In diesem Modus ist auch eine Zusammenarbeit mit den Mikrocontrollern möglich.

Wird bei Reset am PCTL-Eingang High-Pegel festgestellt, ist der 82C08 nachfolgend für alle Operationen im Staus-Modus, d.h. die Eingänge des DRAM-Controllers werden mit den Statusleitungen nach folgender Aufstellung verbunden:

Prozessor	DRAM-Controller
S0	$\overline{WR}$
S1	$\overline{RD}$
S2	PCTL

Die Reaktion des DRAM-Controllers in den verschiedenen Modi zeigt Tabelle 4-26.

PCTL bei Reset:							
0				**1**			
Eingänge			Aktion	Eingänge			Aktion
PCTL	$\overline{RD}$	$\overline{WR}$		PCTL	$\overline{RD}$	$\overline{WR}$	
0	0	0	keine	0	0	0	keine
0	0	1	Lesen	0	0	1	keine
0	1	0	Schreiben	0	1	0	keine
0	1	1	keine	0	1	1	keine
1	0	0	keine	1	0	0	Lesen (Befehl
1	0	1	keine	1	0	1	Lesen (Daten)
1	1	0	keine	1	1	0	Schrieben
1	1	1	keine	1	1	1	keine

Tabelle 4-26. Aktionen des 82C08 als Funktion der Steuereingänge

Die verschiedenen Möglichkeiten einer Hardware-Verbindung mit den Prozessoren zeigen die Abbildungen 4-30, 4-31, 4-32 und 4-33.

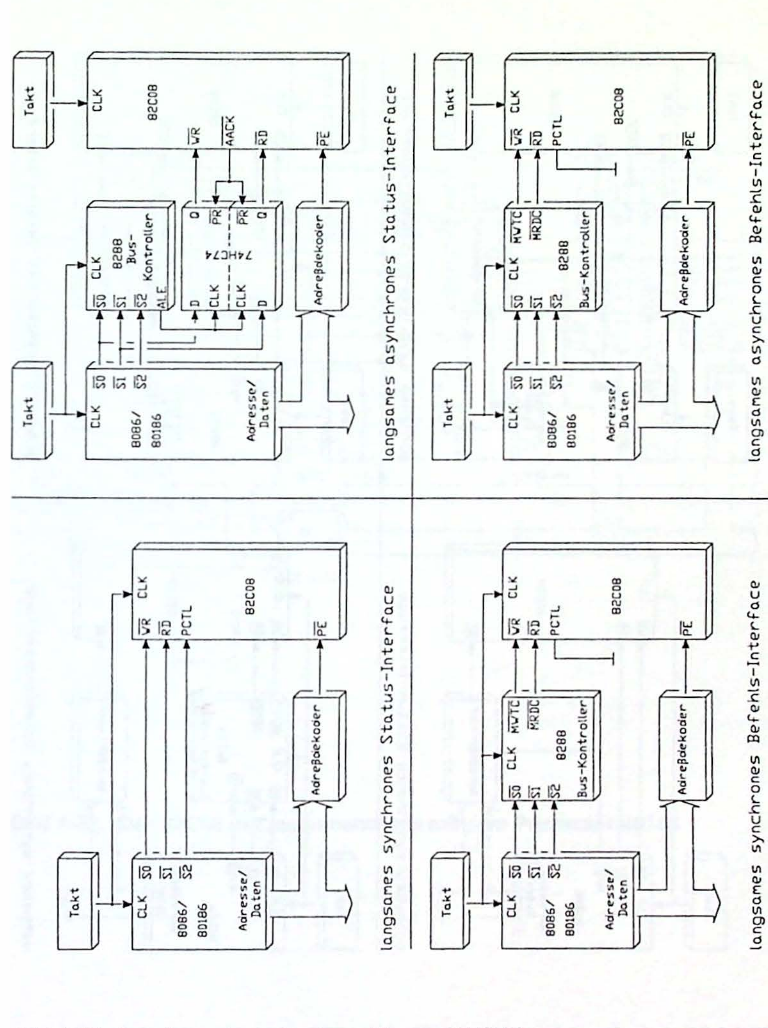


Bild 4-30. Interfaces im langsamen Modus (CFS=0)

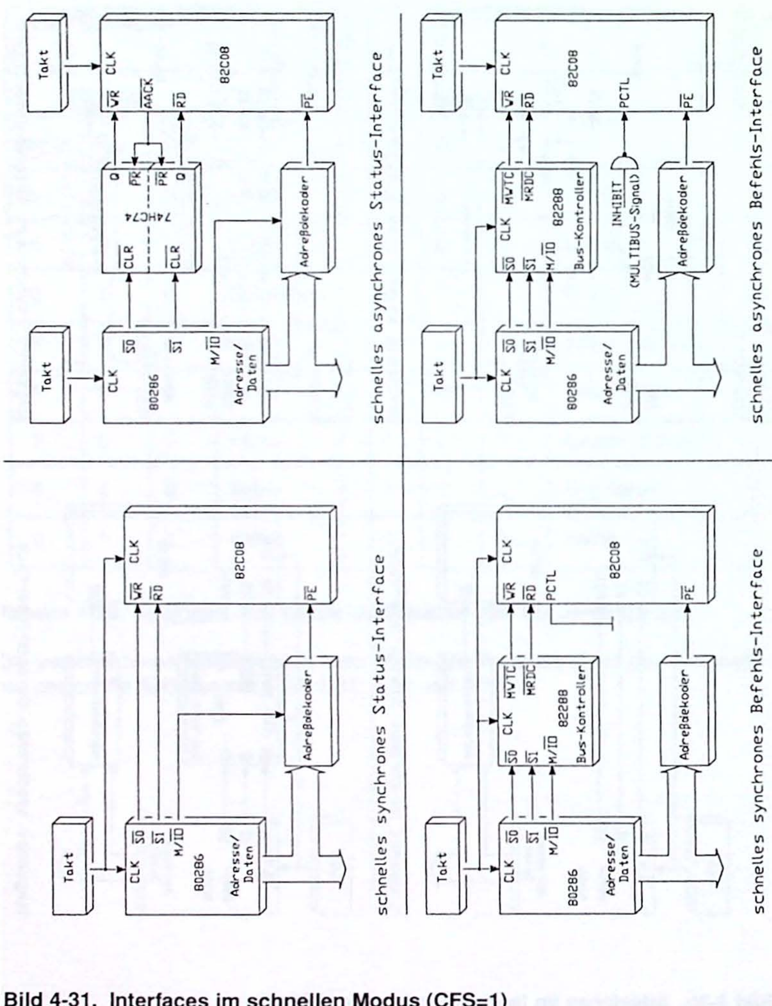


Bild 4-31. Interfaces im schnellen Modus (CFS=1)

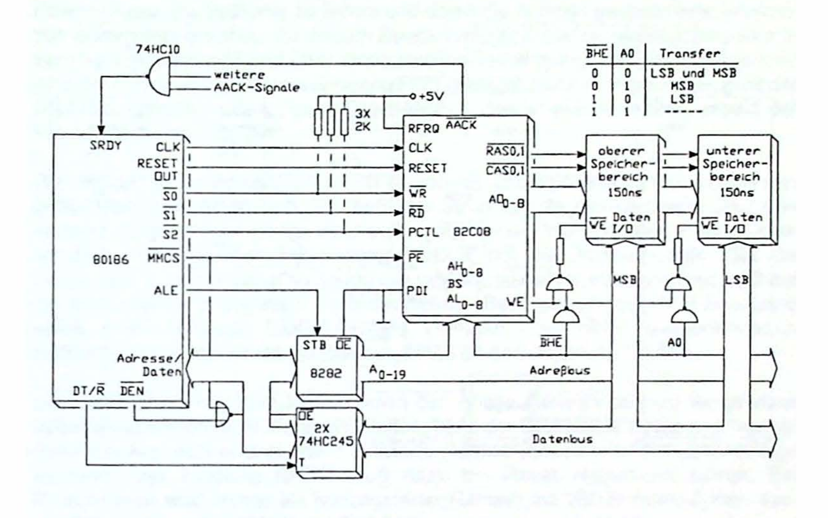


Bild 4-32. Der 82C08 in Zusammenarbeit mit dem Prozessor 80186

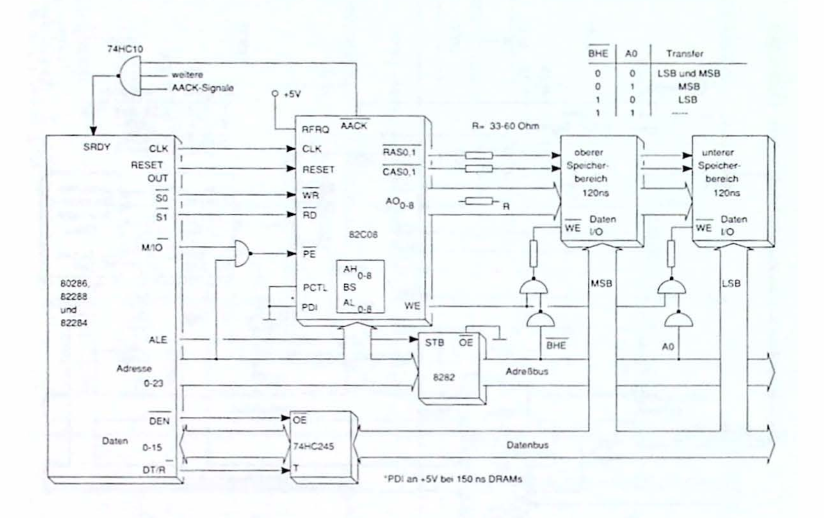


Bild 4-33. Der 82C08 in Zusammenarbeit mit dem Prozessor 80286

4.4.8 Der Power-Down

Ein Hauptgrund für die Beschreibung der CMOS-Version des DRAM-Controllers ist seine Fähigkeit, den nötigen Refresh an die dynamischen RAMs auch bei Power-Down des Systems zu liefern und damit die in ihnen gespeicherte Information aufrecht zu erhalten. Zu diesem Zweck verfügt er über einen separaten Stromversorgungsanschluß und über einen zweiten Takteingang. Bei jeder Taktperiode wird der Power-Down-Detect-Eingang PDD getestet. Führt er High-Pegel, geht der DRAM-Controller in den Power-Down-Modus, den er wieder verläßt, sobald der Pin an Masse zurückkehrt.

Der 82C08 hat zwei Anschlüsse (Pin 24 und 48) für die positive Spannungsversorgung. Über den einen (Pin 48) fließt der Strom für die interne Logik, über den anderen (Pin 24) der Strom für die Ausgabetreiber. Während des Power-Down versorgt ein separater Takteingang PDCLK nur die Refresh-Logik. Um die Leistungsaufnahme dabei so gering wie möglich zu halten, kann die Frequenz auf ein Maß sinken, daß gerade noch die Refresh-Bedingungen der RAM-Bausteine erfüllt sind. Gewisse CMOS-DRAMs verfügen über einen ausgedehnteren Refresh-Zyklus von 64 ms gegenüber 4 ms von gewöhnlichen DRAMs.

Um beim Übergang zum Power-Down das nötige Refresh-Intervall für gewisse Speicheradressen nicht zu überschreiten, kann der DRAM-Controller so programmiert werden, daß er drei intern aktivierte Refresh-Zyklen in unmittelbarer Folge ausführt. Der Eingang RFRQ muß dazu bei Reset High-Pegel führen. Bei Power-Down wird immer ein konzentrierter Refresh mit 256 Refresh-Zyklen ausgeführt, damit auch DRAMs mit 512 Reihen versorgt werden können.

Während des Initialisierungsprozesses geht der 82C08 nicht in den Power-Down. Tritt danach ein Spannungsabfall ein, speichert der 82C08 seinen aktuellen Status und den Inhalt des Refresh-Zählers und führt drei aufeinanderfolgende Block-Refresh-Zyklen aus, sofern RFRQ bei Reset an High-Pegel war. War RFRQ an Masse, erfolgt der Übergang in den Power-Down ohne diese drei Refresh-Zyklen.

Mit der Desaktivierung des Power-Down-Eingangs (PDD an Masse) erzeugt er DRAM-Controller einen internen Reset. Dieser Reset programmiert nicht etwa den Baustein um, noch ändert er den Inhalt des Refresh-Adreßzählers, sondern bewirkt lediglich drei Block-Refresh-Zyklen, die den gesamten Speicher auffrischen.

Das Signal für einen Spannungsabfall muß von einer externen Schaltung erzeugt werden. Der 82C08 testet den PDD-Eingang, sobald die Reset-Programmierung und die RAM-Aufwärmperiode vorüber ist. Bei der Schaltungsentwicklung muß darauf geachtet werden, daß nach Erkennung des Spannungsabfalls durch Wahl eines entsprechenden Netzteils die Systemspannung und der Takt für mindestens 3100 Taktperioden im langsamen Modus und für mindestens 4700 Taktperioden im schnellen Modus stabil erhalten bleiben. Diese Zeit ist nötig, damit der DRAM-Controller die drei konzentrierten Refresh-Zyklen durchführen kann. Wird darauf verzichtet, weil der RFRQ-Pin an Masse liegt, reduziert sich die Zeit auf 20 Systemtakte. Andererseits ist darauf zu achten, daß vor dem Verlassen des Power-Down Spannung und Takt aktiv und stabil sind.

Während des Power-Downs werden vier der Ausgänge nicht aktiviert: $\overline{AACK}$, $\overline{CAS0-1}$ und $\overline{WE}$. Die Adreß- und $\overline{RAS}$-Ausgänge werden benutzt, um den Reihen-Refresh auszuführen. Die $\overline{RAS}$-Leitungen besitzen interne Pull-Ups, um die Leistungsaufnahme der dynamischen RAMs zu reduzieren. Die Stromtreibereigenschaft der Adreßpuffer im Power-Down-Modus ist wesentlich kleiner als bei Normalbetrieb. Dadurch wird die Durchlaufverzögerung der Signale durch die Puffer verlängert. Das ist wichtig, weil dadurch die Stromspitzen auf den Stromversorgungsleitungen im Hinblick auf eine eventuelle schwache Batterieversorgung reduziert werden. Alle Eingänge (außer PDD) sind im Power-Down abgeschaltet.

Der 82C08 verfügt über zwei Takteingänge CLK und PDCLK. Die Taktsignale können derselben Quelle entstammen. Dabei muß allerdings gewährleistet sein, daß der Oszillator auch im Power-Down aktiv ist. Empfehlenswert ist jedoch die Verwendung eines separaten Power-Down-Oszillators, dessen Signal auch im Normalbetrieb am Eingang PDCLK anliegen kann. Bei Spannungsabfall wird der Eingang CLK nach der oben erwähnten Periodenzahl abgeschaltet und der Takt am Eingang PDCLK für den Refresh verwendet. Das Zeitintervall zwischen zwei Refresh-Zyklen beträgt konstant fünf PDCLK-Perioden. Der 82C08 unterstützt die bis auf 64 ms erweiterten Refresh-Zeiten gewisser dynamischer RAMs durch Senkung der PDCLK-Frequenz. Sie kann dabei unter 50 KHz fallen, was eine lange PDCLK-Periode und somit ein ausgedehntes $\overline{RAS}$-Signal zur Folge hat. Der Nachteil, der dabei auftritt, ist der, daß bei Ausdehnung des $\overline{RAS}$-Signals über 1 oder 2 µs CMOS-DRAMs den Strom rapide ansteigen lassen. Um den $\overline{RAS}$-Impuls zu verkürzen, kann man mittels zweier RC-Glieder einen zusätzlichen PDCLK-Impuls einfügen. Die Widerstände und Kondensatoren in Bild 4-34 sollten so gewählt werden, daß t_1 300 ns und t_2 100 ns ist.

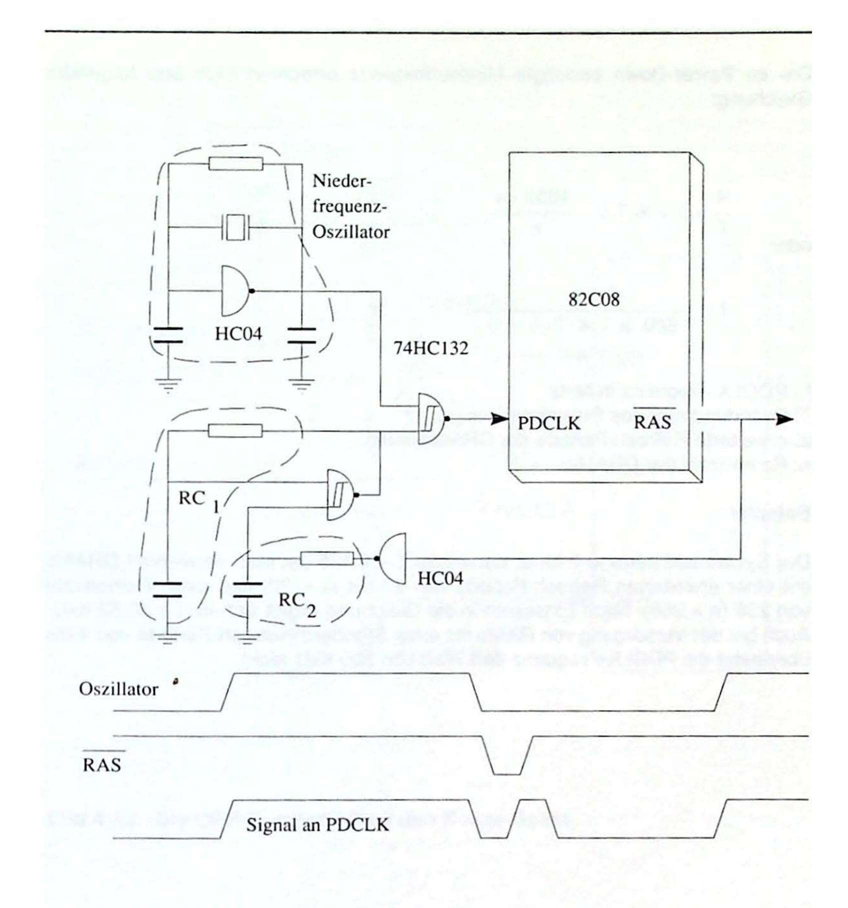


Bild 4-34. Niederfrequenz-Oszillator mit RAS-Signalverkürzung

Die im Power-Down benötigte Mindestfrequenz errechnet sich aus folgender Gleichung:

$$\frac{4}{f} + 1 + 8 \cdot T \leq \frac{1000 \cdot a}{n}$$

oder

$$f \geq \frac{2 \cdot n}{500 \cdot a - 4 \cdot T \cdot n - n}$$

f : PDCLK-Frequenz in MHz
T: Periodendauer des Sytemtaktes in µs
a: erweiterte Refresh-Periode der DRAMs in ms
n: Reihenzahl der DRAMs

Beispiel:

Der Systemtakt betrage 8 MHz, dann folgt T = 0,125 µs. Man verwendet DRAMs mit einer erweiterten Refresh-Periode von 32 ms (a = 32) und einer Reihenzahl von 256 (n = 256). Nach Einsetzen in die Gleichung ergibt sich für f ≥ 32,52 KHz. Auch bei der Versorgung von RAMs mit einer Standard-Refresh-Periode von 4 ms übersteigt die PDCLK-Frequenz den Wert von 300 KHz nicht.

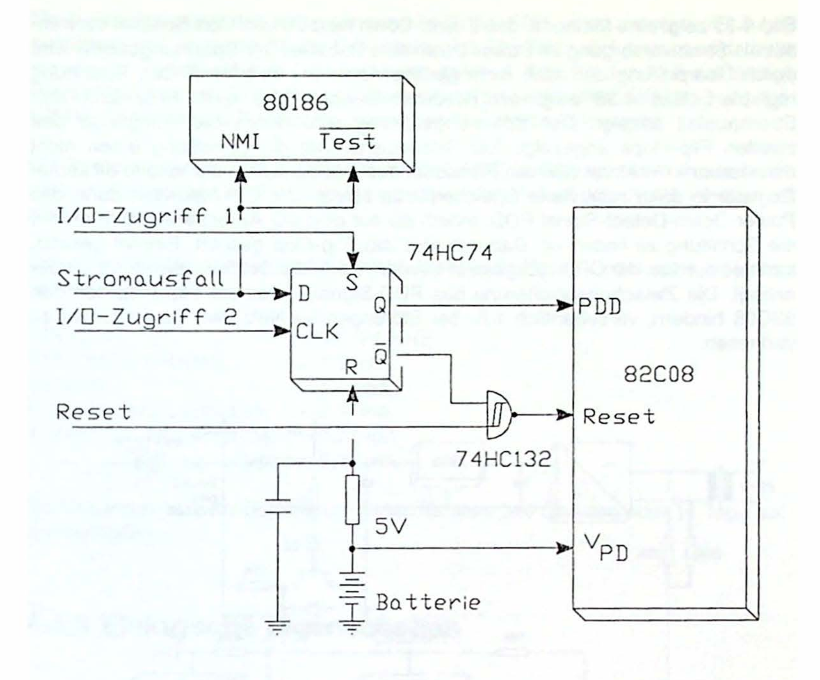


Bild 4-35. Die CPU-Kontrolle über den Power-Down

Bild 4-35 zeigt eine Methode, den Power-Down auszulösen. Das Beispiel verwendet als Stromversorgung im Power-Down eine Batterie. Der Spannungsabfall wird durch Überprüfung der noch nicht gleichgerichteten, transformierten Spannung registriert. Bild 4-36 zeigt ein Hardware-Beispiel, das dem Prozessor den Stromausfall anzeigt. Der Spannungsanstieg wird durch die Ausgänge des zweiten Flip-Flops angezeigt. Bei Stromausfall löst die Schaltung einen nicht maskierbaren Interrupt NMI am Prozessor aus, der daraufhin die Inhalte all seiner Register in dafür reservierte Speicherplätze ablegt. Die CPU aktiviert dann das Power-Down-Detect-Signal PDD, indem sie auf eine I/O-Adresse zugreift, an der die Schaltung zu finden ist. Dadurch wird das Flip-Flop gesetzt. Einmal gesetzt, kann es nur von der CPU rückgesetzt werden, wenn die Betriebsspannung wieder anliegt. Die Zwischenspeicherung des PDD-Signals in einem Flip-Flop soll den 82C08 hindern, versehentlich z.B. bei Störungen im Netz den Power-Down zu verlassen.

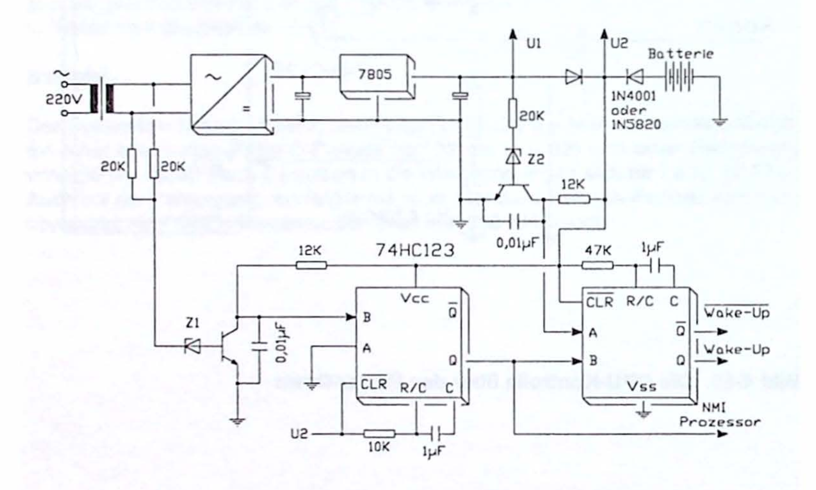


Bild 4-36. Stromausfalldetektor und automatische Wake-Up-Erzeugung bei Spannungsanstieg

Nach Stabilisierung der Netzspannung muß die CPU einen zweiten I/O-Zugriff ausführen, mit dessen Hilfe der Q-Ausgang des Flip-Flops rückgesetzt wird, damit sie wieder Zugriff auf den DRAM-Bereich gewinnt. Das für den NMI zuständige Programm muß im ROM-Bereich stehen und so beschaffen sein, daß es dem DRAM-Controller die Zeit für drei konzentrierte Refresh-Zyklen gibt.

Die Verbindung des $\overline{Q}$-Ausgangs mit einem NAND-Gatter erlaubt ein Reset nur im aktiven Zustand, da ein Reset im Power-Down den Verlust des DRAM-Inhaltes mit sich führen würde.

Das folgende Beispiel zeigt die Stromaufnahme im Power-Down unter typischen Verhältnissen:

f(PDCLK)	= 32,786 KHz
I(16 CMOS DRAMs)	= 1,6 mA
I(82C08)	= 1,0 mA
I(32,786 KHz Oszillator)	= 0,3 mA
I(74HC132, HC74, HC04)	= 0,03 mA
Summe:	2,93 mA

Somit kann eine 1-Ah-Batterie die Daten für etwa 340 Stunden oder 14 Tage aufrechterhalten.

4.4.9 Elektrische Eigenschaften

Die Stromaufnahme beträgt im aktiven Zustand 10 + 2•f mA (f ist die Frequenz in MHz); im Power-Down ist sie 1,0 mA.

Der Baustein ist für die Frequenzen 8, 10, 16 und 20 MHz im Handel. Der Eingangstakt sollte über ein Tastverhältnis von 50 Prozent verfügen.

4.5 Der Port-Baustein 8255

4.5.1 Übersicht

Für jede CPU stellen Ein-/Ausgabe-Ports (im folgenden I/O-Port genannt) eine Notwendigkeit dar. Über sie erfolgt der Kontakt mit der Umwelt und der Datenaustausch mit weiteren Peripheriegeräten. Daten werden an den Drucker gegeben oder die Information von der Computertastatur gespeichert. Meßdaten werden aufgenommen oder Steuerungsaufgaben erfüllt. Ohne einen Ein-/Ausgabe-Port wäre auch das beste Prozessorensystem nur zum Stromverbrauch gut.

I/O-Ports sind also für einen Computer unabdingbar, lediglich die Mikrocontroller verfügen über eigene integrierte Ports. Im Prinzip tut jeder Port-Baustein gute Dienste. Von allen in Frage kommenden Bausteinen ist der 8255 der am häufigsten eingesetzte, weil er durch die Programmierbarkeit und seine einfache Bedienung für die vielfältigsten Aufgaben in Frage kommt. Auch als Port-Erweiterung für die Mikrocontroller ist der Baustein hervorragend geeignet.

Der Baustein 8255 ist in drei Gehäusearten erhältlich. Die Anschlußbelegung zeigt Bild 4-37.

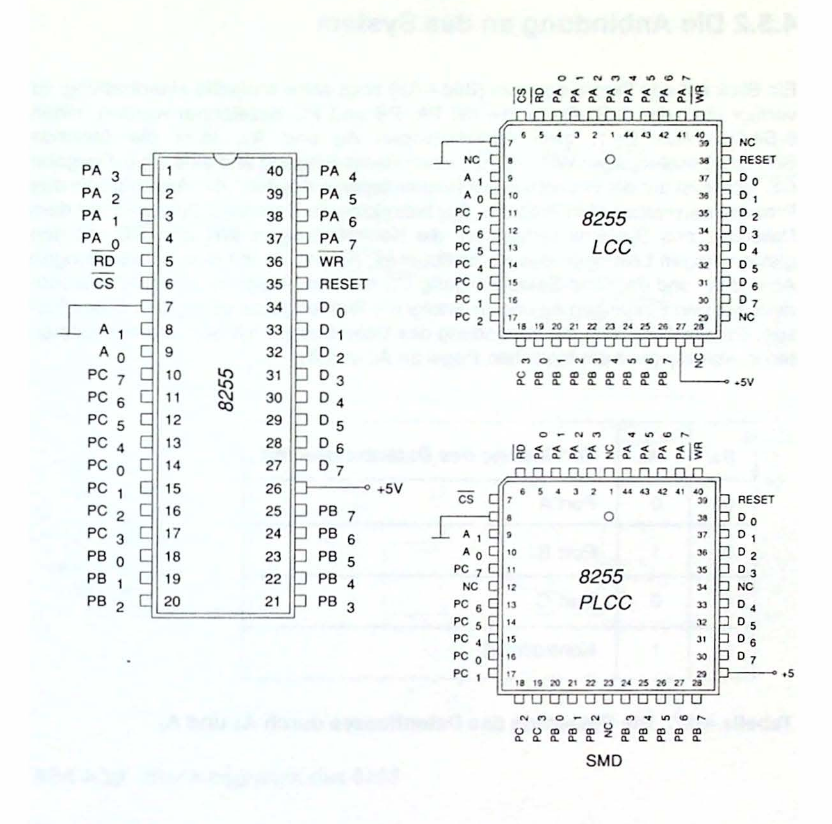


Bild 4-37. Pin-Belegung des 8255

4.5.2 Die Anbindung an das System

Ein Blick auf das Blockdiagramm (Bild 4-38) zeigt seine einfache Handhabung. Er verfügt über drei 8-Bit-Ports, die mit PA, PB und PC bezeichnet werden, einen 8-Bit-Datenbus D_{0-7}, zwei Adreßleitungen A_0 und A_1, über die üblichen Schreib-/Leseleitungen $\overline{WR}$ und $\overline{RD}$, einen Reset-Eingang und eine Chip-Freigabe $\overline{CS}$. Somit ist auf der Grundlage der Buskonzepte in Kapitel 1 der Anschluß an das Prozessorensystem kein Problem. Der bidirektionale Datenbus D_{0-7} wird mit dem Datenbus des Systems verbunden, die Kontrolleitungen $\overline{WR}$ und $\overline{RD}$ mit den gleichnamigen Leitungen des Kontrollbusses, A_0 und A_1 mit den Adreßleitungen A_0 und A_1 und der Chip-Select-Eingang $\overline{CS}$ mit dem Ausgang eines Adreßdekoders. Mit den Eingängen A_0 und A_1 erfolgt die Richtungssteuerung des Datenflusses. Tabelle 4-27 zeigt die Verbindung des Datenbusses mit den internen Einheiten in Abhängigkeit der logischen Pegel an A_0 und A_1.

A_1	A_0	Verbindung des Datenbusses mit
0	0	Port A
0	1	Port B
1	0	Port C
1	1	Kontrollwort

Tabelle 4-27. Die Steuerung des Datenflusses durch A_0 und A_1

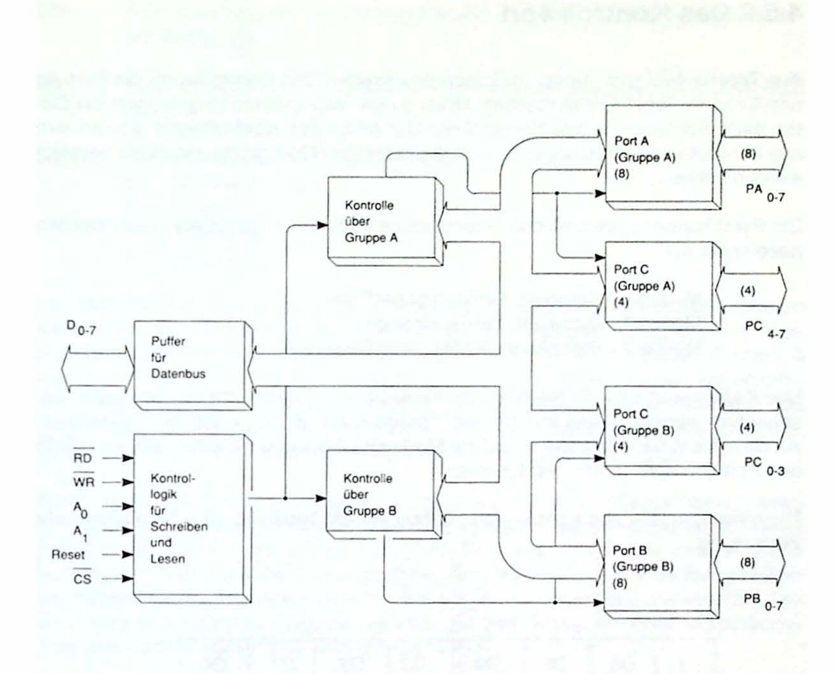


Bild 4-38. Blockdiagramm des 8255

4.5.3 Das Kontrollwort

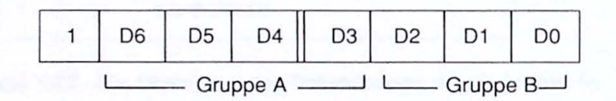
Aus Tabelle 4-27 geht hervor, daß zum Beschreiben des Kontrollworts die Pins A_0 und A_1 High-Pegel führen müssen. Unter dieser Voraussetzung gelangen die Daten beim Schreiben in das Kontrollwort. Der Inhalt des Kontrollworts steuert nun alle Aktivitäten des Bausteins und vor allem die drei Modi, in die die Ports versetzt werden können:

Die Ports können prinzipiell drei verschiedene Funktionen ausüben. Diese werden bezeichnet mit:

Modus 0 - Standard Ein-/Ausgabe-Ports
Modus 1 - Getaktete Ein-/Ausgänge
Modus 2 - Bidirektionaler Bus (nur Gruppe A)

Das Kontrollwort muß für die gewünschte Aktivität mit dem richtigen Bitmuster beschrieben werden. Dabei kommt dem höchsten Bit, Bit 7, besondere Bedeutung zu, da mit ihm das Kontrollwort auf die Moduswahl oder zum Setzen einzelner Bits des Port C umgeschaltet werden kann.

Die einzelnen Bits des Kontrollworts haben zur **Moduswahl** (Bit 7 = 1) folgende Bedeutung:



D0: Eine 1 konfiguriert die unteren vier Bits von **Port C** als Eingänge, eine 0 als Ausgänge.

D1: Eine 1 konfiguriert die acht Bits von **Port B** als Eingänge, eine 0 als Ausgänge.

D2: Eine 1 wählt für die Gruppe B (Port B und die untere Hälfte von Port C) den Modus 1, eine 0 den Modus 0. Modus 2 ist für die Gruppe B nicht möglich.

D3: Eine 1 konfiguriert die oberen vier Bits von **Port C** als Eingänge, eine 0 als Ausgänge.

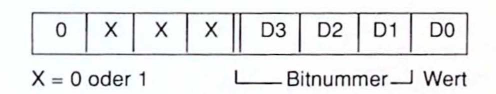
D4: Eine 1 konfiguriert die acht Bits von **Port A** als Eingänge, eine 0 als Ausgänge.

D6,D5: Beide Bits bestimmen den Modus für die Gruppe A:
0 0 : Modus 0
0 1 : Modus 1
1 0 : Modus 2
1 1 : Modus 2

Die Modi können für Gruppe A und Gruppe B unabhängig voneinander definiert werden, so daß die Funktionen der Ports auf die besonderen Bedürfnisse der betreffenden I/O-Struktur zugeschneidert werden können. Zum Beispiel: Gruppe B kann in Modus 0 programmiert werden, um einfache Schalter zu überwachen oder Rechenergebnisse anzuzeigen, während Gruppe A sich in Modus 1 befindet, um eine Tastatur abzufragen oder einen Datenrekorder zu steuern. Die Wirkungsweise der einzelnen Modi ist nachfolgend im Abschnitt 4.5.4 beschrieben.

Wenn Port C als Ausgang benutzt wird, kann jeder seiner acht Pins einzeln gesetzt oder gelöscht werden. Besondere Bedeutung kommt dieser Tatsache zu, wenn Port C in den Modi 1 und 2 Status- bzw. Kontrollfunktionen für die Ports A und B ausübt. Der Zugriff auf die einzelnen Pins erfolgt ebenfalls durch ein Beschreiben des Kontrollworts. Allerdings muß in diesem Fall darauf geachtet werden, daß das Bit 7 eine Null aufweist, denn nur so kann der 8255 zwischen einer Modusänderung und dem **Setzen eines Bits** unterscheiden.

Nur die unteren vier Bits des Kontrollwort haben dann eine Bedeutung:



D0: Bit Null stellt den an den Pin zu schreiben Wert dar. 0 heißt Ausgabe von Low-, 1 Ausgabe von High-Pegel.

D3,D2,D1: Diese drei Bits stellen die binärkodierte Nummer des zu beschreibenden Pins von Port C dar. Dabei gilt die Zuordnung nach Tabelle 4-28:

D3	D2	D1	ausgewählter Port-Pin
0	0	0	PC_0
0	0	1	PC_1
0	1	0	PC_2
0	1	1	PC_3
1	0	0	PC_4
1	0	1	PC_5
1	1	0	PC_6
1	1	1	PC_7

Tabelle 4-28. Das Bit-Set/Reset-Format (D7=0)

Soll beispielsweise Pin 2 von Port C auf High-Pegel gesetzt werden, muß das Bitmuster 0000 0101_b (=05_h) in das Kontrollwort geschrieben werden. Das Bitmuster 0101 1100_b (=$5C_h$) bewirkt Low-Pegel an Pin 6. Beide Beispiele setzen voraus, daß Port C zuvor durch Moduswahl als Ausgang konfiguriert wurde.

Das Kontrollwort ist lesbar. Beim Lesen erfährt man den Inhalt, der mit Bit7 = 1 in das Kontrollwort geschrieben wurde, also Informationen über den eingestellten Modus. Bit 7 ist immer eine Eins.

4.5.4 Die Ports

4.5.4.1 Modus 0

Der häufigste Einsatz der Ports liegt in Modus 0 als Ein-/Ausgabe-Ports. Diese Benutzung ist am einfachsten zu handhaben.

Nach einem positiven Impuls am Reset-Eingang sind alle Ports als **Eingänge** geschaltet. Werden die Ports in diesem Fall durch die CPU gelesen, erscheinen alle Pins als Einsen. Sie befinden sich dabei im Tri-State, so daß keine Strombelastung des Eingangssignals auftritt. In Modus 0, dem Standard I/O-Modus, sind die intern vorhandenen Eingangsspeicher allerdings nicht aktiv, die Signale müssen in der Zeit anliegen, in der sie von der CPU gelesen werden.

Will man die Ports oder Teile davon aus **Ausgänge** benutzen, ist nach einem Reset das Beschreiben des Kontrollworts nötig. Die Information, die die CPU an den betreffenden Port schreibt, wird dort gespeichert. Darin sind sich alle drei Ports gleich. Die Stromtreibereigenschaften der Ports sind nicht besonders hoch. Jeder Pin kann einen Strom von typisch 2,5 mA fließen lassen, wobei eine höhere Belastung bis zu 8 mA dem Baustein nicht schadet. Somit kann also eine LED ohne Schutzwiderstand angesteuert werden.

Wie dem Blockdiagramm (Bild 4-38) zu entnehmen ist, bilden die Ports A und B für sich immer eine Einheit von acht Bits, die entweder als Ein- oder Ausgänge konfiguriert sind. Lediglich Port C ist in zwei Hälften von je vier Bits trennbar, so daß aus den drei Ports vier Port-Einheiten entstehen können. Jede Port-Einheit kann unabhängig von der anderen durch Beschreiben der Bits D_{7-0} des Kontrollworts als Eingang oder Ausgang definiert werden. Somit sind 16 verschiedene I/O-Kombinationen möglich (Tabelle 4-29).

$D_7 = 1; D_6, D_5, D_2 = 0$				Gruppe A		Gruppe B	
D_4	D_3	D_1	D_0	Port A	Port C_{4-7}	Port B	Port C_{0-3}
0	0	0	0	Ausgang	Ausgang	Ausgang	Ausgang
0	0	0	1	Ausgang	Ausgang	Ausgang	Eingang
0	0	1	0	Ausgang	Ausgang	Eingang	Ausgang
0	0	1	1	Ausgang	Ausgang	Eingang	Eingang
0	1	0	0	Ausgang	Eingang	Ausgang	Ausgang
0	1	0	1	Ausgang	Eingang	Ausgang	Eingang
0	1	1	0	Ausgang	Eingang	Eingang	Ausgang
0	1	1	1	Ausgang	Eingang	Eingang	Eingang
1	0	0	0	Eingang	Ausgang	Ausgang	Ausgang
1	0	0	1	Eingang	Ausgang	Ausgang	Eingang
1	0	1	0	Eingang	Ausgang	Eingang	Ausgang
1	0	1	1	Eingang	Ausgang	Eingang	Eingang
1	1	0	0	Eingang	Eingang	Ausgang	Ausgang
1	1	0	1	Eingang	Eingang	Ausgang	Eingang
1	1	1	0	Eingang	Eingang	Eingang	Ausgang
1	1	1	1	Eingang	Eingang	Eingang	Eingang

Tabelle 4-29. Die 16 möglichen Port-Konfigurationen in Modus 0

Die Eigenschaften der Ports in Modus 0 auf einen Blick:

- Zwei 8-Bit- und zwei 4-Bit-Ports.
- Jeder Port kann unabhängig als Aus- oder Eingang konfiguriert werden.
- Die Ausgabewerte bleiben gespeichert.
- Die Eingänge besitzen keine Speicher.
- Es sind 16 verschiedene I/O-Konfigurationen möglich.

4.5.4.2 Modus 1

Dieser Modus wird gewählt, wenn ein I/O-Datentransfer mit peripheren Bausteinen in Verbindung mit Kontroll- oder Quittierungssignalen erfolgen soll.

Die besonderen Eigenschaften der Ports in diesem Modus sind:

- Die Ports sind in zwei Gruppen aufgeteilt, Gruppe A und Gruppe B.
- Jede Gruppe besteht aus einem 8-Bit-Daten-Port und einem 4-Bit-Kontroll-Port.
- Der Daten-Port kann entweder Ein- oder Ausgang sein.
- Sowohl die Aus- als auch die Eingänge besitzen interne Speicher.
- Der 4-Bit-Port wird für Kontrollzwecke oder Statusmeldung über den Daten-Port benutzt.

Die Daten-Ports sind also bis auf die Eingangsspeicher genau so beschaffen wie in Modus 0. Besondere Bedeutung kommt den jeweiligen Pins von Port C zu, die nun Verwaltungsaufgaben erfüllen. Dabei ist die Art ihrer Funktion davon abhängig, ob der Port im Kontrollwort als Ein- oder Ausgang konfiguriert wurde.

4.5.4.2.1 Die Ports als Eingänge

Port C wird zur Versorgung der Gruppen A und B nicht in der Mitte getrennt (Bild 4-39). Die Bits PC_{0-2} gehören zur Gruppe B, die Bits PC_{3-5} zur Gruppe A. Pro Gruppe sind also nur drei Steuerleitungen vorhanden. Zwei Pins von Port C (PC_6 und PC_7) bleiben dabei unbenutzt und können als I/O-Pins für allgemeine Zwecke benutzt werden. Ihre Konfigurierung erfolgt mit Bit D_3 des Kontrollworts.

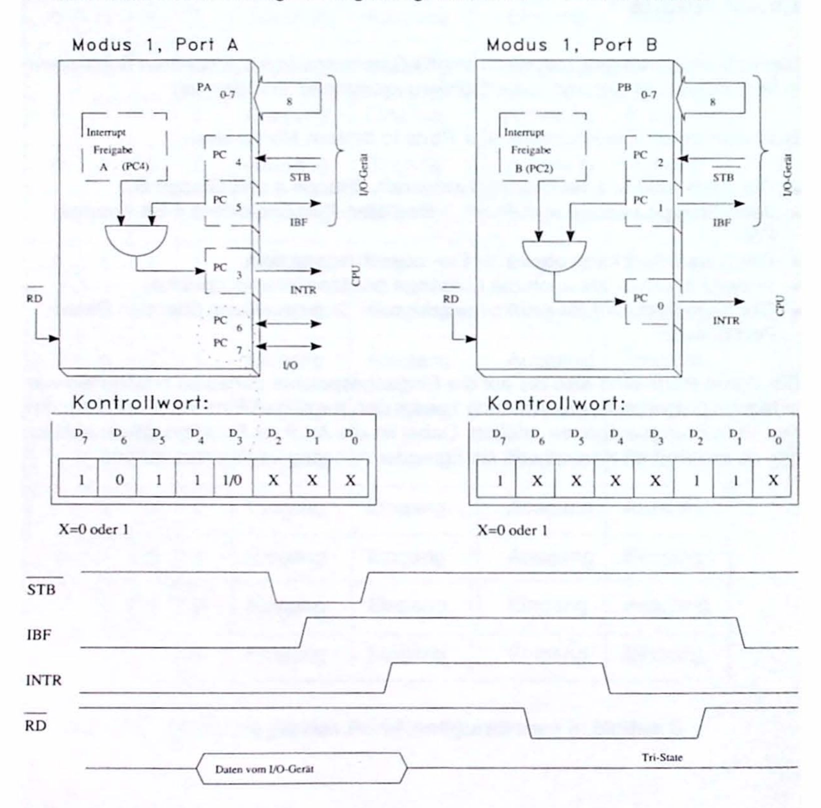


Bild 4-39. Die Ports in Modus 1 als Eingänge konfiguriert

Sollen nun Daten in Port A oder B geschrieben werden, bedarf es eines Taktsignals, das die internen Eingangsspeicher des 8255 veranlaßt, die anliegenden Daten zu übernehmen. Diesem Zweck dient das $\overline{STB}$-Signal (Strobe). Der Eingang $\overline{STB}$ ist nicht flanken-, sondern pegelgetriggert, d.h. Masse an diesem Eingang schreibt die Daten in die Speicher. Das die Daten ausgebende Gerät muß damit auch Sorge tragen, daß das $\overline{STB}$-Signal zum richtigen Zeitpunkt, d.h. unmittelbar nach Ausgabe seiner Daten, an den 8255 gelangt.

Da im allgemeinen einige Zeit vergeht, bis die CPU die eingegangenen Daten gelesen hat, muß der Port ein weiteres Beschreiben verhindern, da sonst die ersten Daten überschrieben werden könnten. Aus diesem Grund gibt der 8255 über den Ausgang IBF (Input Buffer Full) High-Pegel aus, den alle schreibwilligen Peripheriegeräte benutzen können, so lange Wartezyklen auszuführen, bis die CPU durch einen Lesezugriff auf den 8255 den High-Pegel der IBF-Leitung löscht. Mit Masse auf der IBF-Leitung ist nun erneutes Beschreiben der Ports erlaubt.

Um nun die Verweildauer der Daten in den Ports möglichst gering zu halten, kann der 8255 so programmiert werden, daß mit dem Beschreiben des Ports gleichzeitig ein Interrupt bei der CPU ausgelöst wird. Im Prinzip könnte man das IBF-Signal dazu benutzen; es läßt sich nur nicht abschalten. Aus diesem Grund stellt der 8255 ein Interrupt-Signal INTR zur Verfügung, das freigegeben oder gesperrt werden kann. Die Steuerung erfolgt durch ein internes Flip-Flop, das mit einem Bit-Setz-Befehl für Port C (D_7 des Kontrollworts ist 0) gesetzt oder gelöscht werden kann. Zur Interrupt-Freigabe durch Port A muß das Bit PC_4, zur Interrupt-Freigabe durch Port B das Bit PC_2 gesetzt sein. Das gleiche gilt für das Sperren des Interrupts. Das Bitmuster 0000 1001 in das Kontrollwort geschrieben gibt also den Interrupt durch Port A frei. Das Setzen der Einzelbits wirkt sich in Modus 1 also nicht auf die logischen Zustände der Pins von Port C aus, sondern setzt interne Flip-Flops. Ähnlich wie das IBF-Signal wird der aktive Pegel des INTR-Signals durch ein Lesen des Ports durch die CPU gelöscht.

4.5.4.2.2 Die Ports als Ausgänge

Werden die Ports in Modus 1 durch Beschreiben des Kontrollworts als Ausgänge definiert, üben sechs Pins von Port C ebenfalls Kontrollfunktionen aus. Man beachte jedoch, daß im Falle der Gruppe A andere Pins zur Kontrolle verwendet werden im Vergleich zur Eingangsfunktion (Bild 4-40). Nicht benötigt werden die Pins PC_4 und PC_5, die für allgemeine I/O-Zwecke zur Verfügung stehen. Über ihre Funktion entscheidet Bit D_3 im Kontrollwort.

Sobald die CPU Daten in den Port geschrieben hat, legt der 8255 die $\overline{OBF}$-Leitung (Output Buffer Full) an Masse, um einerseits einem I/O-Gerät anzuzeigen, daß Daten anliegen, und andererseits die CPU wissen zu lassen, daß ein I/O-Gerät die Daten noch nicht gelesen hat. Durch Testen dieser Leitung kann die CPU feststellen, ob sie die nächsten Daten an den 8255 senden kann. Einfacher geht es mit der INTR-Leitung. Mit Beschreiben eines Ports geht auch diese Leitung an Masse. Beim Lesen der Daten durch ein I/O-Gerät geht nicht nur die $\overline{OBF}$-.Leitung, sondern auch die INTR-Leitung an High. Der High-Pegel kann nun bei der CPU einen Interrupt auslösen, so daß das ständige Abfragen der $\overline{OBF}$-Leitung entfällt. Im Prinzip könnte man auch die $\overline{OBF}$-Leitung zur Interrupt-Erzeugung heranziehen; dieser Interrupt wäre allerdings nicht abschaltbar.

Der Ausgang INTR wird durch ein internes Flip-Flop gesteuert. Mit dem Bit-Setz-Befehl kann es gelöscht oder gesetzt werden. Um den Interrupt in Gruppe A freizugeben, muß das Bit PC_6 gesetzt, für eine Freigabe in Gruppe B das Bit PC_2 gesetzt werden. Auch hier hat wiederum der Zugriff auf die Einzelbits keine Wirkung auf die logischen Pegel der entsprechenden Pins. Ist der Interrupt unterbunden, führt der Ausgang INTR immer Masse.

Ein I/O-Gerät bekommt mittels Low-Pegel an der $\overline{OBF}$-Leitung die Mitteilung, daß Daten zum Lesen anstehen. Sobald es bereit ist, die Daten aufzunehmen, legt es die Leitung $\overline{ACK}$ (Acknowledge) an Masse. Das bewirkt, daß der $\overline{OBF}$-Pegel von Minus nach Plus wechselt. Viele Bausteine benötigen diese positive Flanke, um damit die Daten in ihren Speicher zu takten. Während des ganzen Vorgangs behält die INTR-Leitung ihren Low-Pegel bei. Erst nachdem die $\overline{ACK}$-Leitung wieder High-Pegel führt, geht auch die INTR-Leitung an Plus, um der CPU durch einen Interrupt anzuzeigen, daß der Lesevorgang durch das I/O-Gerät abgeschlossen ist.

Eine Kombination von Ein- oder Ausgabefunktion beider Ports ist jederzeit möglich, so daß eine parallele bidirektionale 8-Bit-Datenkommunikation möglich ist.

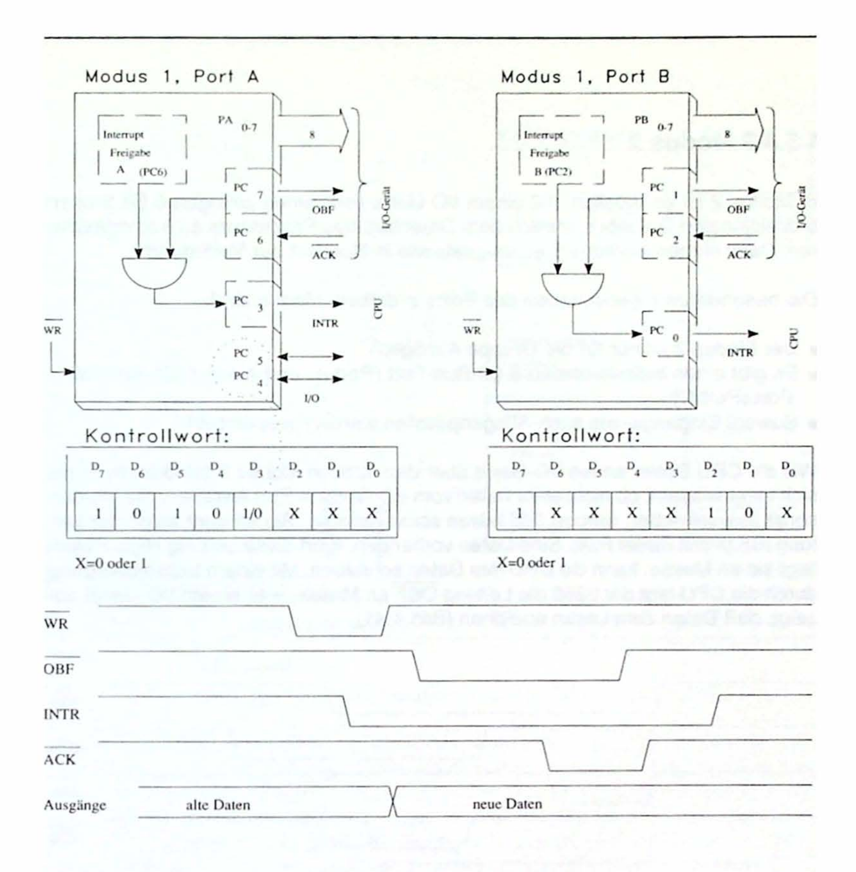


Bild 4-40. Die Ports in Modus 1 als Ausgänge konfiguriert

4.5.4.3 Modus 2

In Modus 2 ist es möglich, mit einem I/O-Gerät über einen einzigen 8 Bit breiten bidirektionalen Datenbus ähnlich dem Datenbus des Prozessors zu kommunizieren. Dazu stehen ähnliche Steuersignale wie in Modus 1 zur Verfügung.

Die besonderen Eigenschaften des Ports in diesem Modus sind:

- Der Modus 2 ist nur für die Gruppe A möglich.
- Es gibt einen bidirektionalen 8-Bit-Bus-Port (Port A) und einen 5-Bit-Kontroll-Port (Port C).
- Sowohl Eingangs- als auch Ausgangsdaten werden gespeichert.

Will die CPU Daten an ein I/O-Gerät über den 8255 in Modus 2 senden, muß sie sich vergewissern, ob nicht etwa Daten vom I/O-Gerät in Port A stehen. Sie würden sonst überschrieben werden und wären somit verloren. Sie erkennt es an der Leitung IBF (Input Buffer Full). Sind Daten vorhanden, führt diese Leitung High-Pegel; liegt sie an Masse, kann die CPU ihre Daten schreiben. Mit einem Schreibvorgang durch die CPU legt der 8255 die Leitung $\overline{OBF}$ an Masse, was einem I/O-Gerät anzeigt, daß Daten zum Lesen anstehen (Bild 4-41).

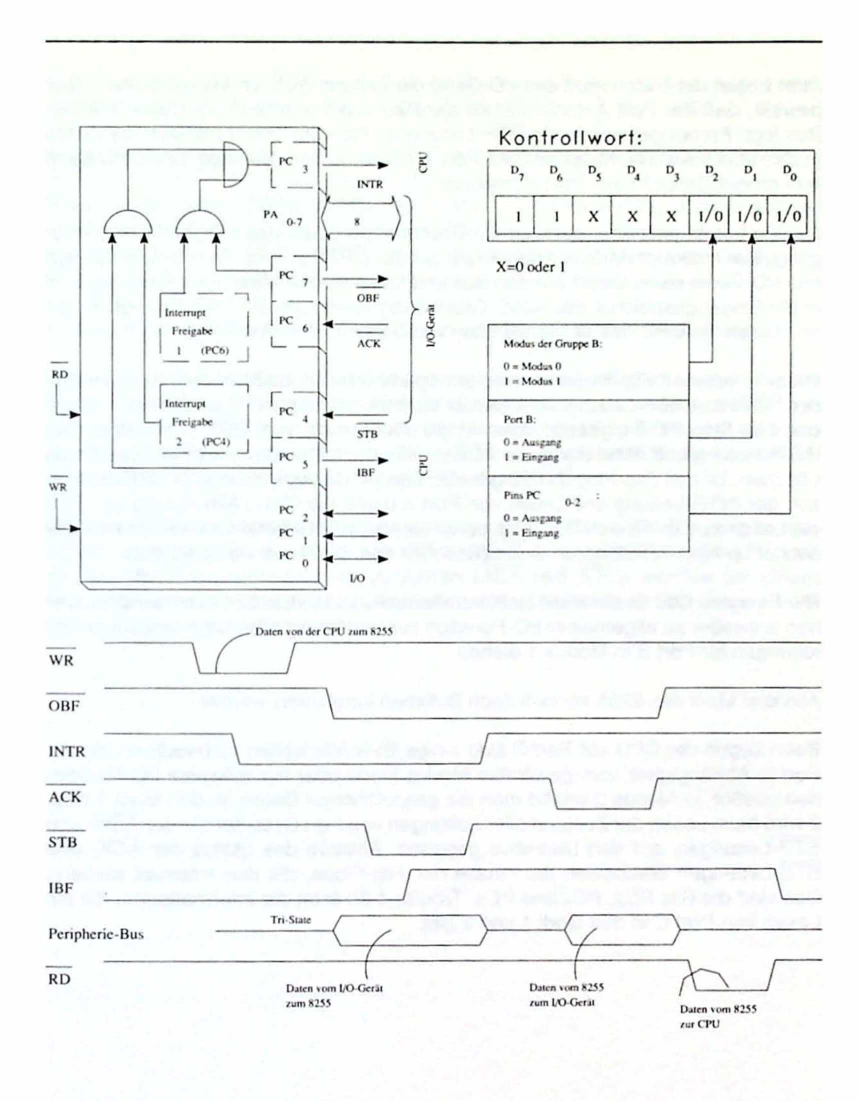


Bild 4-41. Port A in Modus 2

Zum Lesen der Daten muß das I/O-Gerät die Leitung $\overline{ACK}$ an Masse ziehen. Das bewirkt, daß der Port A den Tri-State der Pins abschaltet und die Daten auf den Bus legt. Ferner geht dabei die $\overline{OBF}$-Leitung an Plus und taktet dadurch die Daten in das I/O-Gerät. Nachfolgend geht Port A wieder in den Tri-State. Die CPU kann nun erneut Daten in den Port schreiben.

Der Port steht jetzt aber auch für ein Beschreiben durch das I/O-Gerät zur Verfügung. Der Indikator dafür ist High-Pegel auf der $\overline{OBF}$-Leitung. Zum Schreiben legt das I/O-Gerät seine Daten auf den Bus und taktet sie mit Masse am Eingang $\overline{STB}$ in die Eingangsspeicher des 8255. Gleichzeitig nimmt die IBF-Leitung High-Pegel an, woran die CPU das Vorhandensein von Daten in Port A erkennt.

Mit zwei internen Flip-Flops kann es ermöglicht werden, daß mit dem Aktivwerden der $\overline{OBF}$- bzw. IBF-Leitung ein Interrupt über die Leitung INTR ausgelöst wird. Ist das Flip-Flop INTE1 gesetzt, bewirkt die Aktivierung von $\overline{OBF}$ ein Setzen der INTR-Leitung und Aktivierung von $\overline{ACK}$ (Lesen der Daten durch das I/O-Gerät) ein Löschen. Ist das Flip-Flop INTE2 gesetzt, bewirkt die Aktivierung von IBF ein Setzen der INTR-Leitung und Lesen von Port A durch die CPU (Aktivierung von $\overline{RD}$) ein Löschen. Flip-Flop INTE1 kann durch einen Bit-Setz-Befehl an das Bit PC_6 gesetzt, Flip-Flop INTE2 durch einen Befehl an das Bit PC_4 gesetzt werden.

Die Pins des Port C, die nicht für Kontrollzwecke in Modus 2 benutzt werden, stehen entweder zu allgemeiner I/O-Funktion zur Verfügung oder können als Kontrollleitungen für Port B in Modus 1 dienen.

Alle drei Modi des 8255 können nach Belieben kombiniert werden.

Beim Zugriff der CPU auf Port C sind einige Besonderheiten zu beachten, da der Port in Abhängigkeit vom gewählten Modus keine oder nur teilweise I/O-Funktionen ausübt. In Modus 0 erfährt man die gespeicherten Daten. In den Modi 1 bzw. 2 wird beim Lesen der Zustand aller Leitungen von Port C, außer die der $\overline{ACK}$- und $\overline{STB}$-Leitungen, auf den Datenbus gegeben. Anstelle des Status der $\overline{ACK}$- und $\overline{STB}$-Leitungen erscheinen die Inhalte der Flip-Flops, die den Interrupt steuern. Das sind die Bits PC_2, PC_4 und PC_6. Tabelle 4-30 listet die Informationen, die ein Lesen von Port C in den Modi 1 und 2 gibt.

Modus:	D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
Modus 1, Eingang	I/O	I/O	IBF_A	$INTE_A$	$INTR_A$	$INTE_B$	IBF_A	$INTR_B$
Modus 1, Ausgang	$\overline{OBF}_A$	$INTE_A$	I/O	I/O	$INTR_A$	$INTE_B$	$\overline{OBF}_B$	$INTR_B$
Modus 2	$\overline{OBF}_A$	$INTE_1$	IBF_A	$INTE_2$	$INTR_A$	Modus der Gruppe B		

Tabelle 4-30. Informationen beim Lesen von Port C in den Modi 1 und 2

Mit einem Schreibbefehl an Port C können nur die Pins beeinflußt werden, die als Ausgänge in Modus 0 konfiguriert sind. Es können keine anderen Pins dadurch geändert werden und schon gar nicht die internen Interrupt-Flip-Flops. Auf sie kann nur mit einem Bit-Setz-Befehl zugegriffen werden. Das gilt auch für die als Kontrollleitungen nicht benutzten restlichen I/O-Pins des Port C.

Mit diesem Bit-Setz-Befehl kann jede Leitung von Port C, die als Ausgang festgelegt ist, einschließlich der Ausgänge IBF und $\overline{OBF}$, beschrieben werden. Die Leitungen, die Eingangsfunktionen ausführen ($\overline{ACK}$ und $\overline{STB}$), werden bei einem Schreibbefehl nicht tangiert; an ihrer Stelle werden die internen Flip-Flops zur Interrupt-Steuerung beschrieben.

4.5.5 Reset

High-Pegel am Eingang Reset löscht den Inhalt des Kontrollregisters und setzt alle Ports in Modus 0 als Eingänge. Die Port-Pins sind somit im Tri-State. Die Wartezeit, nach der der 8255 nach einem Power-On-Reset zu benutzen ist, beträgt mindestens 50 µs. Nachfolgende Resetimpulse benötigen nur noch eine Zeit von 500 ns (=0,5 µs).

4.5.6 Anwendungsbeispiele

4.5.6.1. Modus 0

Bild 4-42 gibt ein Beispiel für den Anschluß an die CPU 8085 wieder. Der in diesem Bild gezeigte Adreßdekoder weist dem 8255, der in Modus 0 betrieben wird, die Adressen 00 bis 03 zu. Durch Beschreiben des Kontrollworts mit dem Wert 1001 0000 (90_h) wird Modus 0 eingestellt und die Ports B und C als Ausgänge, Port A als Eingang konfiguriert:

```
 MVI  A,90#

OUT   03#  ; Daten an das Kontrollwort
```

Mit diesen beiden Befehlen wird der 8255 initialisiert. Um nachfolgend Daten von Port A zu lesen und Daten an die Ports B und C zu schreiben, können die drei folgenden Befehle benutzt werden:

```
 IN   00#  ; Inhalt von Port A --> CPU Akkumulator
OUT   01#  ; Akkumulator --> Port B
OUT   02#  ; Akkumulator --> Port C
```

Nachdem der Modus 0 gesetzt ist, arbeitet jeder Port wie ein normaler Port.

Um Daten aus Port A zu lesen sowie Daten an Port B zu schreiben und den Pin 1 von Port C zu setzen, kann nachstehende Befehlsfolge benutzt werden:

```
IN   00#    ; Inhalt von Port A --> CPU Akkumulator
OUT  01#    ; Akkumulator --> Port B
MVI  A,03#  ; Bit-Setz-Befehl für PC1
OUT  03#    ; Beschreiben des Kontrollworts
```

Die anderen Pins des Port C werden in diesem Fall nicht verändert.

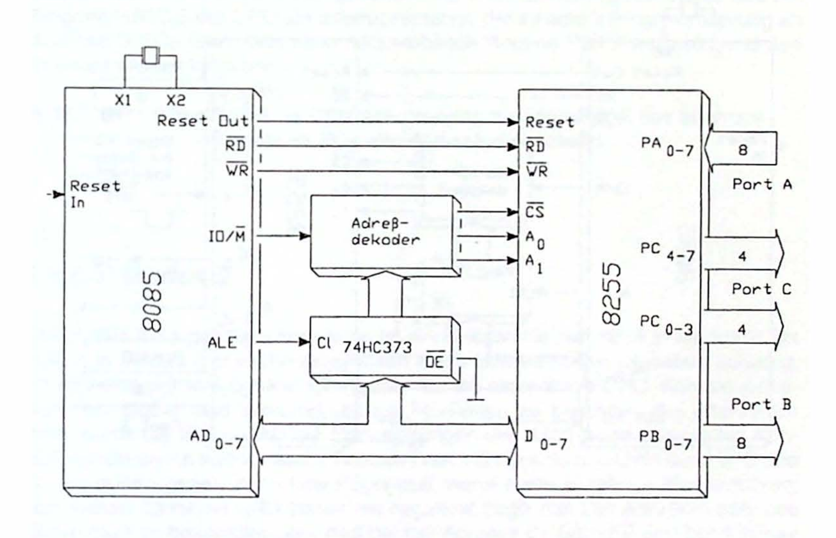


Bild 4-42. Die Verwendung des 8255 in Modus 0

4.5.6.2. Modus 1

Ein Beispiel, das den 8255 in Modus 1 in Zusammenarbeit mit der CPU 8085 zeigt, ist in Bild 4-43 dargestellt.

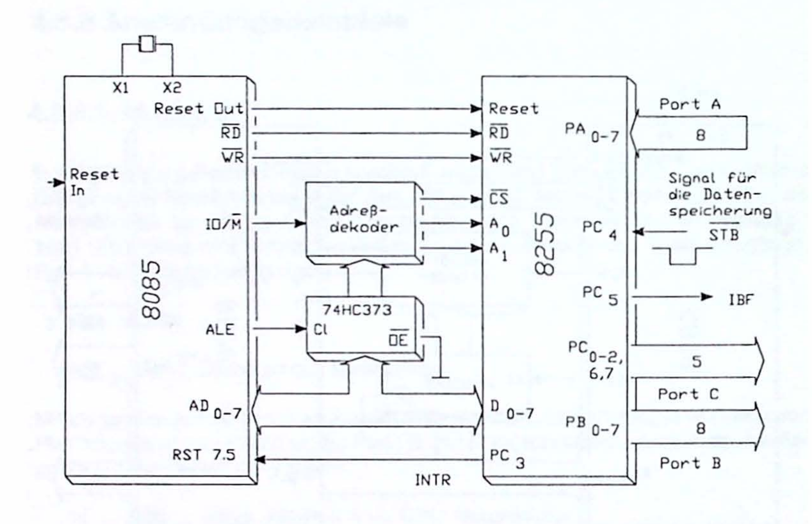


Bild 4-43. Anwendungsbeispiel des 8255 für Modus 1

Port A wird in Modus 1 als Eingang für die getaktete Datenaufnahme verwendet. Ein I/O-Gerät sendet seine Daten an Port A und gibt ein Taktsignal an PC_4. Dadurch werden die Daten in die internen Speicher des Port A geschrieben. Der 8255 setzt den Pin PC_5 (IBF Input Buffer Full). Wenn zuvor ein Bit-Setz-Befehl an PC_4 erfolgte, kann an der CPU über Pin PC_3 (INTR) ein Interrupt ausgelöst werden. Der in Bild 4-43 nicht gezeigte Port B kann entweder im gleichen Modus oder in Modus 0 arbeiten.

Das zu Bild 4-43 gehörende Programm lautet:

```
MVI   A,B0# ; In das Kontrollwort wird der Wert 1011 0000 geschrieben.
              Somit ist Port A in Modus 1 Eingang; die anderen Ports sind
              Ausgänge
OUT    03#  ; Akkumulator --> Kontrollwort
MVI   A,09# ; Bit PC4 wird gesetzt, um
OUT    03#  ; einen Interrupt zu ermöglichen
EI          ; Interrupt-Freigabe
HLT         ; Stop der Aktivitäten
```

Sobald das I/O-Gerät die Daten gesendet und in die Speicher getaktet hat, wird am Eingang RST7,5 der CPU ein Interrupt erzeugt, der zu einem Programmsprung an Adresse 003Ch führt. Dort kann nachstehende Routine Port A auslesen und den Interrupt wieder freigeben:

```
003C  IN    00# ; Port A --> CPU Akkumulator, der High-Pegel des Interrupt-
                  Signals an PC3 wird dadurch rückgesetzt.
      EI
      RET
```

4.5.6.3. Modus 2

Die in Bild 4-44 gezeigte Schaltung ist ein Beispiel für den 8255 in Modus 2. Da Port A in Modus 2 alle Charakteristiken eines bidirektionalen Datenbus aufweist, ist es keine größere Schwierigkeit, über ihn mit einer Slave-CPU Kontakt aufzunehmen. Dabei sind allerdings einige Feinheiten zu beachten, die erforderlich sind, damit die Slave-CPU die Statusleitungen des 8255 lesen bzw. selbst Kontrollsignale an ihn senden kann. Diesem Zweck dienen die drei OR-Gatter B, C und D. Sie geben genau dann Low-Pegel aus, wenn beide Eingänge Masse führen, und stellen damit ein AND-Gatter mit negativer Logik dar. Der Adreßdekoder des Slave muß so beschaltet sein, daß bei der Adresse 01 Gatter B und bei Adresse 00 die Gatter C und D freigegeben sind. Dann haben die Befehle der Slave-CPU folgende Wirkung:

IN 01#:
Das ist ein Lesebefehl an den Port A des 8255. Der Adreßdekoder sperrt unter der Adresse 01 die Gatter C und D und gibt Gatter B frei. Gleichzeitig geht der $\overline{RD}$-Pin der Slave-CPU an Masse. Dadurch bleibt der Ausgang des Port A im Tri-State ($\overline{ACK}$ wird ja nicht aktiviert), und lediglich die beiden Datenpuffer 74HC125 geben die Signale der Ausgänge IBF und $\overline{OBF}$ an die Datenleitungen D_0 und D_1, womit sie für die CPU lesbar sind.

IN 00#:
Bei dieser Adresse gibt der Adreßdekoder die Gatter C und D frei und sperrt nur das Gatter B. Da die CPU gleichzeitig das $\overline{RD}$-Signal aktiviert, wird nur Gatter C durchgeschaltet, dessen Ausgang mit dem $\overline{ACK}$-Eingang verbunden ist. Dadurch legt der 8255 die intern gespeicherten Daten des Port A auf den Slave-Datenbus.

OUT 00#:
Bei diesem Befehl wird die $\overline{WR}$-Leitung der Slave-CPU aktiv. Zusammen mit dem Dekodersignal führt der Ausgang des Gatters D Low-Pegel und taktet damit über den Eingang $\overline{STB}$ die auf dem Slave-Datenbus anstehenden Daten in den Port A des 8255, von wo aus sie durch die Master-CPU in der üblichen Weise lesbar sind.

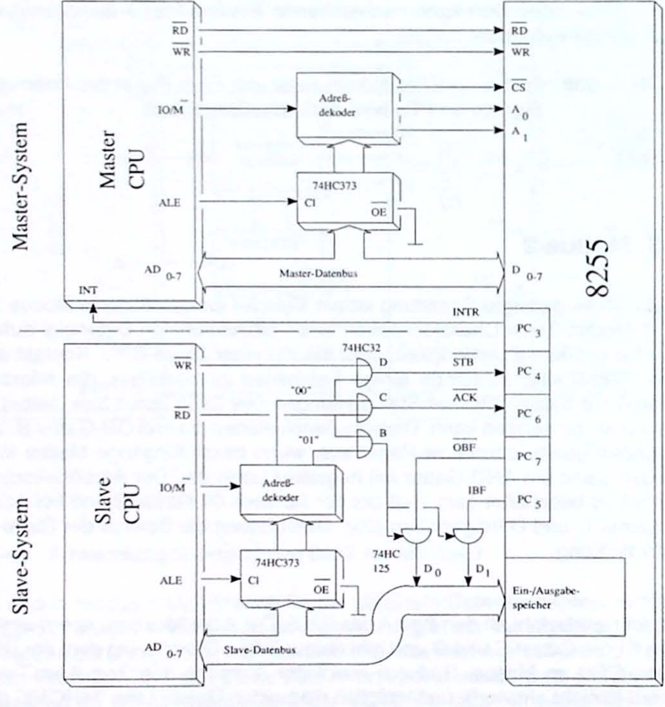


Bild 4-44. Die Verwendung des 8255 in Modus 3

Bei einer Kommunikation dieser Art sollten folgende Schritte eingehalten werden:

1. Der 8255 wird durch die Master-CPU in Modus 2 versetzt.

2. Die Master-CPU schreibt die Daten, die an den Slave gerichtet sind, in gewohnter Weise an Port A. Daraus folgt Low-Pegel am Ausgang $\overline{OBF}$.

3. Die Slave-CPU ließt entweder kontinuierlich die Statusleitungen IBF und $\overline{OBF}$ oder läßt sich mit Masse an $\overline{OBF}$ durch einen Interrupt unterbrechen (z.B. Mikrocontroller-Familie MCS-51).

4. Sobald der Slave Low-Pegel am Ausgang $\overline{OBF}$ feststellt, gibt er einen Lesebefehl an Adresse 00 aus. Dadurch werden die Ausgänge des Port A aktiviert und das Signal $\overline{OBF}$ auf High-Pegel gesetzt.

5. Während dieser Periode ließt die Master-CPU den Status des 8255, d.h. Port C, und überprüft insbesondere die Bits 5 (IBF) und 7 ($\overline{OBF}$). Solange $\overline{OBF}$ Masse führt, bedeutet das, daß die Daten vom Slave noch nicht gelesen wurden und der Master keine neuen Daten an Port A schreiben kann. Erst wenn das Bit 7 eine Eins aufweist, kann der Master die nächsten Daten senden.

6. Wenn die Daten zur Master-CPU fließen sollen, muß sie die Slave-CPU an den Port A des 8255 mit der Adresse 00 schreiben. Dadurch nimmt der Ausgang IBF High-Pegel an.

7. Die Master-CPU liest den Status, d.h. Port C, und überprüft das dazugehörende IBF-Bit. Ist der Inhalt eine Eins, liegen für den Master Daten zum Lesen an. Mit einem Lesebefehl an Port A fließen die Daten in den Master und das IBF-Signal wird gelöscht.

8. Die Slave-CPU überwacht ihrerseits durch Lesen der Adresse 01 den Status der IBF-Leitung. Solange sie High-Pegel aufweist, können keine neuen Daten in den 8255 geschrieben werden. Erst Masse zeigt an, daß der Master die alten Daten aus Port A entnommen hat.

9. Auf diese Art können Daten ausgetauscht werden. Da Port A über zwei getrennte Speicher verfügt (Eingabe- und Ausgabespeicher), ist es nicht nötig, den Ein-/Ausgabetransfer abwechselnd durchzuführen.

Dieser Ablauf kann dadurch vereinfacht werden, daß die Interrupt-Leitung INTR den Master bei Dateneingang durch den Slave unterbricht, so daß die ständige Abfrage der Statusleitungen entfällt.

Vier weitere Anwendungsmöglichkeiten in den verschiedenen Modi zeigt Bild 4-45.

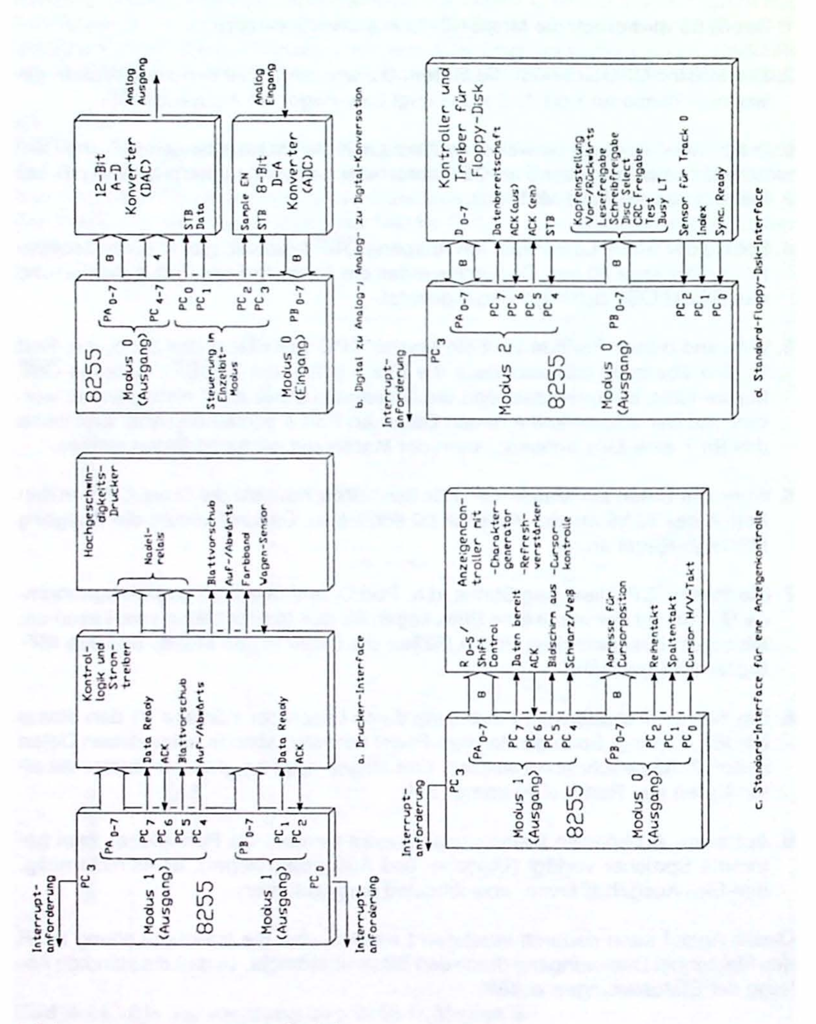


Bild 4-45. Anwendungsbeispiele

4.5.7 Elektrische Eigenschaften

Spannungswerte an den Eingängen, die oberhalb +2 V liegen, erkennt der 8255 als High-Pegel, Spannungen unterhalb +0,8 V als Low-Pegel. Bei der Ausgabe von Signalen liegt der High-Pegel oberhalb +2,4 V und Low-Pegel unterhalb +0,4 V. Die Stromaufnahme beträgt maximal 120 mA.

Die Zugriffszeiten über den bidirektionalen Datenbus betragen in der langsamsten Version beim Lesen 300 ns, beim Schreiben 400 ns. Die restlichen Zeiten der Steuersignale in den Modi 1 und 2 können stark mit der Qualität des Bausteins und von Hersteller zu Hersteller schwanken, so daß bei zeitkritischen Anwendungen Einblick in die entsprechenden Datenblätter genommen werden muß.

Die CMOS-Version 82C55 ist im Hinblick auf Kompatibilität den Spannungswerten für die logischen Pegel der NMOS-Version angepaßt. Unterschiede liegen in der Stromaufnahme (10 µA) und in der Zugriffszeit. Letztere beträgt beim 82C55 etwa 120 ns.

4.5.8 Der zweifache Port-Baustein 82C255

Eng verwandt mit dem 8255 ist der zweifache Port-Baustein 82C255 , der nur in der CMOS-Version erhältlich ist.

Kontaktadresse: Mitshubishi Electric Europe
Gothaer Straße 6
4030 Ratingen 1

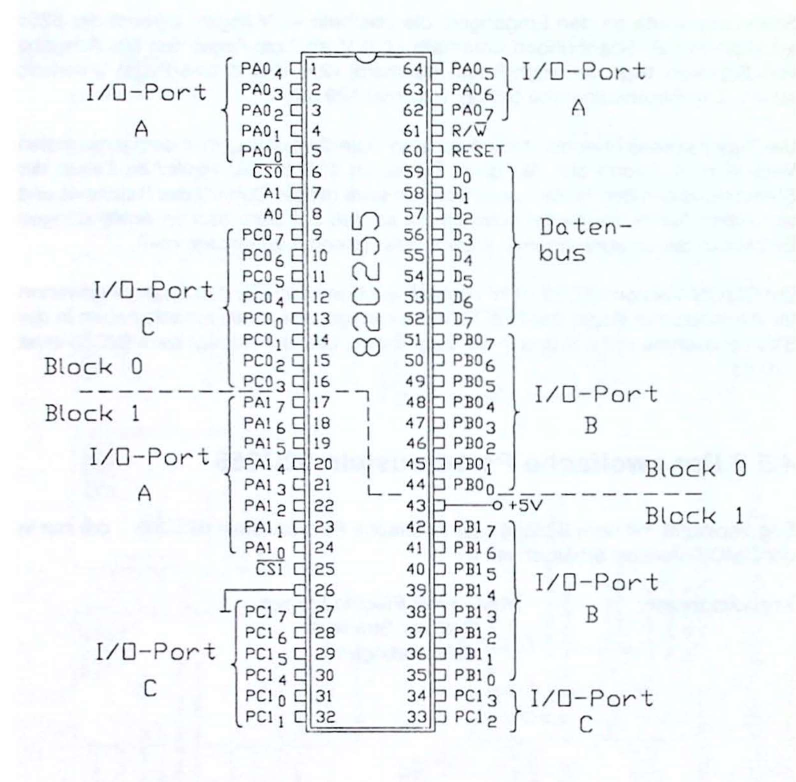


Bild 4-46. Die Pin-Belegung des 82C255

Der 82C255 ist bis auf zwei kleine Unterschiede nichts anderes als zwei 82C55-Bausteine in einem Gehäuse integriert. Dadurch wird der Tatsache Rechnung getragen, daß ein übliches Prozessorensystem einen großen Bedarf an I/O-Ports aufweist und häufig mehr als nur einen 82C55 benötigt. Durch die Integration zweier Einzelbausteine wird die Zahl der Ports verdoppelt, wie in Bild 4-46 zu sehen ist. Man kann ihn in zwei identische Blöcke, Block 0 und Block 1, unterteilen, wobei jeder die gleichen elektrischen und logischen Eigenschaften wie der 82C55 aufweist. Die bidirektionalen Busleitungen und die Steuerleitungen sind nur einmal vorhanden und wirken auf beide Blöcke gleichzeitig (Bild 4-47).

Die Auswahl des betreffenden Blocks erfolgt mit Hilfe der beiden Leitungen $\overline{CS0}$ und $\overline{CS1}$. Führt $\overline{CS0}$ Masse, ist die CPU mit Block 0 verbunden, führt $\overline{CS1}$ Masse, beziehen sich alle Zugriffe des Prozessors auf Block 1. Eine interne *Exklusiv-Oder*-Schaltung verhindert die Bausteinselektierung, wenn beide Eingänge gleichzeitig Masse führen.

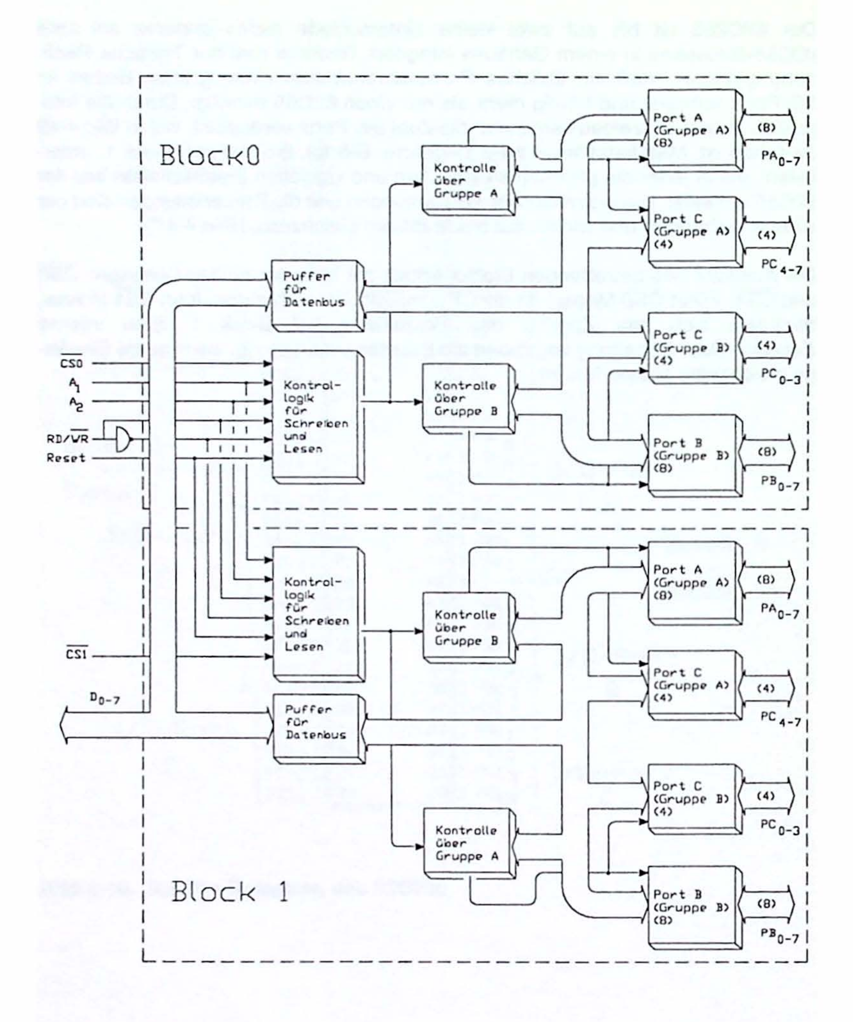


Bild 4-47. Das Blockdiagramm des 82C255

Der zweite Unterschied besteht in der Verwendung des Schreib-/Lesesignals von der CPU. Der 82C255 besitzt zum Schreiben und Lesen denselben Pin, Pin Nr. 61. Liegt an ihm High-Pegel, wird der selektierte Block gelesen, führt er Masse, werden die auf dem Bus stehenden Daten in ihn geschrieben. Man könnte nun meinen, daß das low-aktive $\overline{RD}$-Signal zum Lesen invertiert an diesen Eingang geführt werden müßte. Das ist nicht nötig. Man bedenke, daß nach Selektierung und beim Lesen des Bausteins die $\overline{WR}$-Leitung ja an Plus bleibt, also genau das tut, was ein invertiertes $\overline{RD}$-Signal bewirken würde. Der 82C255 verhält sich wie viele Speicherbausteine, die über keinen eigenen Output-Enable-Eingang verfügen. Er legt die durch A0 und A1 adressierten Daten auf den Bus, wenn das $\overline{WR}$-Signal ausbleibt und ein aktives $\overline{CS}$-Signal vorliegt. Für die Praxis bedeutet das: man verbindet lediglich die $\overline{WR}$-Leitung mit dem 82C255, die $\overline{RD}$-Leitung bleibt frei.

Da der 82C255 in seinen weiteren Eigenschaften identisch mit denen des 8255 ist, ist die vorstehende Beschreibung des 8255 ohne Einschränkungen für jeden Block des 82C255 gültig.

4.6 Der Kommunikationsbaustein 8250

4.6.1 Die serielle Datenübertragung

Es gibt zahlreiche Fälle, in denen ein Computersystem einen Datenaustausch mit peripheren Bausteinen wünscht, ihn aber mangels Verbindungsleitungen nicht parallel ausführen kann. Beispiele dieser Art sind die Informationen, die das System von der Tastatur erhält. In der Tastatur sitzt im Falle eines Personal Computers ein Mikrocontroller, meist ein Vertreter der MCS-48-Familie, dessen Aufgabe nur darin besteht, die Tastatur zu überwachen und den Kode einer gedrückten Taste an das System zu melden. Um diesen 8-Bit-Kode zu übertragen, ist nur ein Kabel mit wenigen Adern vorhanden. Es können also nicht alle acht Bits des Bytes gleichzeitig (parallel) übermittelt werden, sondern nur nacheinander (seriell).

Ähnliche Verhältnisse trifft man bei der Datenübermittlung einer Maus an das System an. Hier fließen die Daten von der Maus ebenfalls seriell über die RS-232-C-Schnittstelle in den Rechner.

Wachsende Bedeutung genießt jedoch die Datenübertragung beim Datenaustausch von zwei Computern untereinander über die vorhandenen seriellen Schnittstellen, wobei hier allerdings mit größerem Aufwand auch eine parallele Vernetzung möglich ist. Der Nachteil einer seriellen Datenübertragung liegt in der Reduzierung der Geschwindigkeit des Datenaustausches. Er muß dort in Kauf genommen werden, wo es auf andere Art nicht möglich ist, z.B. bei der Datenfernübertragung über das Telefonnetz. Hier begrenzen noch zur Zeit die Länge der Leitungen und die analoge Technik die Übertragungsgeschwindigkeit. Die Geräte, die die Verbindung von digitalen Daten zum Telefonnetz herstellen, werden Modems genannt. Generell spielt die Übertragungsgeschwindigkeit eine entscheidende Rolle bei dieser Art der Datenübertragung. Die Zahl der Bits, die in einer Sekunde auf seriellem Wege übertragen werden, wird Baudrate genannt.

Weitere Probleme liegen in der Synchronisation von Sender und Empfänger. Bei jedem Übertragungsvorgang sollten Sender und Empfänger sich über die verwendete Baudrate einig sein. Darüber hinaus gibt es insbesondere bei langen Verbindungsleitungen Störeinflüsse, die zu einer Verfälschung der Daten führen können und die erkannt werden müssen. Aus diesem Grund werden ein oder mehrere Bits einem Datum (Datum ist Singular von Daten) angehängt, aus denen der Empfänger auf einen Fehler schließen kann.

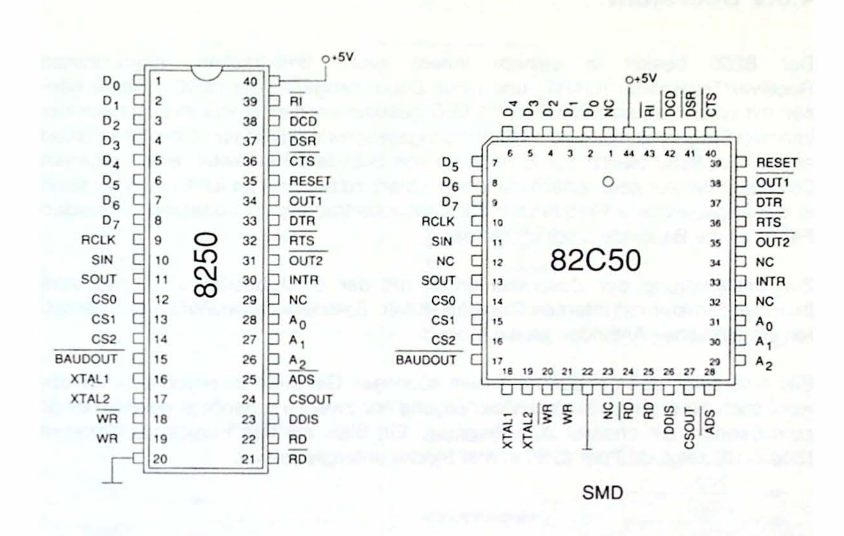


Bild 4-48. Die Pin-Belegung des 8250

Da der ganze Vorgang nach demselben Schema stereotyp abläuft, stellen Bausteine, die auf diese Aufgabe hin konzipiert wurden, eine wesentliche Entlastung der CPU und des Programmierers dar. Einer davon ist der 8250. Er ist ein universeller, asynchroner Receiver/Transmitter (UART) mit integriertem **B**aud**r**aten**g**enerator (BRG).

4.6.2 Übersicht

Der 8250 besitzt in seinem Innern einen universellen, asynchronen Receiver/Transmitter (UART) und einen Baudratengenerator (BRG). Beide können mit einer Frequenz von 0 bis 10 MHz getaktet werden, wodurch sich nach der internen Frequenzteilung eine Übertragungsgeschwindigkeit von 0 bis 625 KBaud ergibt. Der 8250 besitzt zur Erzeugung von Standard-Baudraten einen eigenen Oszillator, der nur noch extern mit einem Quarz zu beschalten ist. Er ist aber auch in der Lage, externe Frequenzen zur Datenübertragung zu benutzen. In beiden Fällen ist die Baudrate programmierbar.

Zur Erleichterung der Zusammenarbeit mit der CPU besitzt der 8250 eine Interrupt-Struktur mit internen Prioritätsstufen. Spezielle Kontrolleitungen gestatten ein einfaches Anbinden an ein Modem.

Bild 4-48 zeigt, daß der 8250 in einem 40poligen Gehäuse untergebracht ist, obwohl doch für die serielle Datenübertragung nur zwei Pins benötigt werden: einer zum Senden, ein anderer zum Empfang. Ein Blick auf das Funktionsdiagramm (Bild 4-49) zeigt, daß der 8250 in drei Blöcke untergliedert ist.

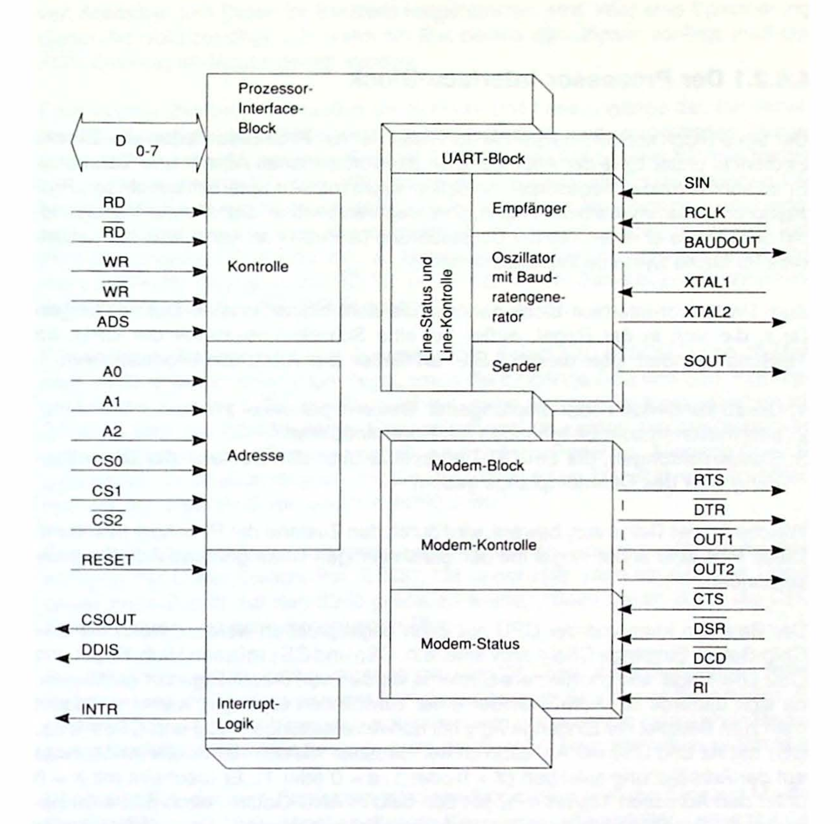


Bild 4-49. Das Funktionsdiagramm des 8250

4.6.2.1 Der Prozessor-Interface-Block

Der erste Block mit den meisten Anschlüssen ist der **Prozessor-Interface-Block**. Er dient in erster Linie der Anbindung an den vorhandenen Adreß- und Datenbus. Er ist sehr luxuriös ausgestattet, so daß er leicht mit den unterschiedlichsten Prozessoren zusammenarbeiten kann. Des weiteren stellt er Signale zur Verfügung, mit deren Hilfe er einen Teil der Bussteuerung übernehmen kann, was ihn besonders für kleine Systeme interessant macht.

Zum Prozessor-Interface-Block gehören die acht bidirektionalen Datenleitungen D_{0-7}, die sich in der Regel, außer bei eine Schreib-/Lesezugriff der CPU, im Tri-State befinden. Über diesen 8-Bit-Port fließen drei Arten von Informationen:

1. Die zu sendenden oder empfangenen Daten in paralleler Form;
2. Informationen zum Beschreiben der Kontrollregister;
3. Statusmeldungen, die der CPU Aufschluß über den Zustand der Datensendung oder des Datenempfangs geben.

Welche Art der Daten sich bewegt, wird durch den Zustand der Pins A_{0-2} bestimmt. Diese Pins sind in der Regel mit den gleichnamigen Leitungen des Adreßbusses verbunden.

Der Baustein kann von der CPU nur dann angesprochen werden, wenn die drei Chip-Select-Eingänge CS_{0-2} aktiv sind, d.h. CS_0 und CS_1 müssen High-Pegel und $\overline{CS2}$ Low-Pegel führen. Kleinere Systeme werden von dieser Eigenart profitieren, da sich dadurch ein Adreßdekoder unter Umständen erübrigen kann. Verbindet man zum Beispiel die Eingänge A_{0-2} mit den Adreßleitungen A_{0-2} und CS_0 mit A_3, CS_1 mit A_4 und $\overline{CS3}$ mit A_5, dann ist der Baustein mit dem Bitmuster XX01 1aaa auf der Adreßleitung selektiert (X = 0 oder 1; a = 0 oder 1). Er erscheint mit X = 0 unter den Adressen 18_h bis $1F_h$. Mit den beiden AND-Gattern nach Bild 4-50 besitzt er die Adressen $F0_h$ bis $F7_h$. Würde man anstelle der AND-Gatter NOR-Gatter verwenden, wäre der Adreßbereich von 00_h bis $0F_h$.

Ein weiterer Pin kann die Anbindung an einen gemultiplexten Adreß-/Datenbus erleichtern, es ist der Address-Strobe-Eingang $\overline{ADS}$. Mit seiner Hilfe können logischen Pegel an den Eingängen A_{0-2} und CS_{0-2} im 8250 gespeichert werden, so daß sie nicht für den gesamten Schreib-/Lesevorgang stabil an diesen Eingängen anliegen müssen. Die an diesen Pins anliegenden Daten werden mit der positiven Flanke des $\overline{ADS}$-Signals intern gespeichert. Das ALE-Signal des Prozessors kann invertiert zum Takten an diesen Eingang geführt werden, so daß ein Demultiplexen

von Adressen und Daten im Baustein vorgenommen wird. Wird eine Speicherung dieser Art nicht benötigt, z.B. wenn der Bus bereits demultiplext vorliegt, muß der $\overline{\text{ADS}}$-Eingang an Masse gelegt werden.

Eine weitere Besonderheit stellen die Schreib- und Leseeingänge dar, mit denen die CPU Zugriff auf den 8250 nimmt. Sie sind jeweils paarweise vorhanden, einmal für positive, einmal für negative Schreib-/Lesesignale. Für positive Signale tragen sie die Bezeichnung RD bzw. WR, für negative Signale die Bezeichnungen $\overline{\text{RD}}$ bzw. $\overline{\text{WR}}$. Zu beachten ist dabei, daß für einen Zugriff nur ein Eingang, nicht beide, aktiv ist. Üblicherweise wird das negative $\overline{\text{RD}}$-Signal verwendet. In diesem Fall muß der Eingang RD inaktiv, d.h. an Masse bleiben. Benutzt man zur Steuerung das positive RD-Signal, ist der $\overline{\text{RD}}$-Pin an Plus zu halten. Das gleiche gilt ohne Einschränkungen für die $\overline{\text{WR}}$-/WR-Eingänge.

Für gewisse Zwecke kann der Ausgang Chip Select Out (CSOUT) von Nutzen sein. Er führt genau dann High-Pegel, wenn die Eingänge CS_0 und CS_1 High-Pegel und der Eingang $\overline{\text{CS2}}$ Low-Pegel führen, wenn also der Baustein selektiert ist. Dadurch kann die Adreßdekodier-Eigenschaft genutzt werden, um beispielsweise andere Bausteine am gemeinsamen Bus durch High-Pegel am Ausgang CSOUT zu sperren. Es ist auch möglich, mit diesem Signal einen Transceiver freizuschalten, der den Datenfluß von und zum 8250 puffert.

Ein weiterer Ausgang, der sich insbesondere in kleinen Systemen nützlich machen kann, ist der Driver-Disable-Pin (DDIS). Mit seiner Hilfe kann ein externer Transceiver beim Zugriff auf den 8250 gesteuert werden. Beim Lesen durch die CPU führt er Masse und beim Beschreiben Plus.

Zur Erleichterung der Kommunikation mit der CPU besitzt der 8250 einen Interrupt-Ausgang INTR, der von einer komplexen inneren Struktur gesteuert wird. Er geht an Plus, wenn der Interrupt freigegeben wird und eine von zahlreichen Bedingungen erfüllt ist. Näheres findet sich bei der Beschreibung der Interrupt-Struktur.

Der letzte Pin, der zum Prozessor-Interface-Block gehört, ist der Eingang Master Reset (MR). Liegt High-Pegel an diesem Pin, werden alle seriellen Datenaktivitäten abgebrochen. Das Modem-Kontrollregister wird zusammen mit den entsprechenden Ausgängen gelöscht und die verschiedenen internen Register mit Standardwerten beschrieben. Um weiterarbeiten zu können, muß der Eingang an Low-Pegel zurückkehren. Der Eingang verfügt über eine Schmitt-Trigger-Charakteristik, so daß auch langsam ansteigende Signale verwendet werden können.

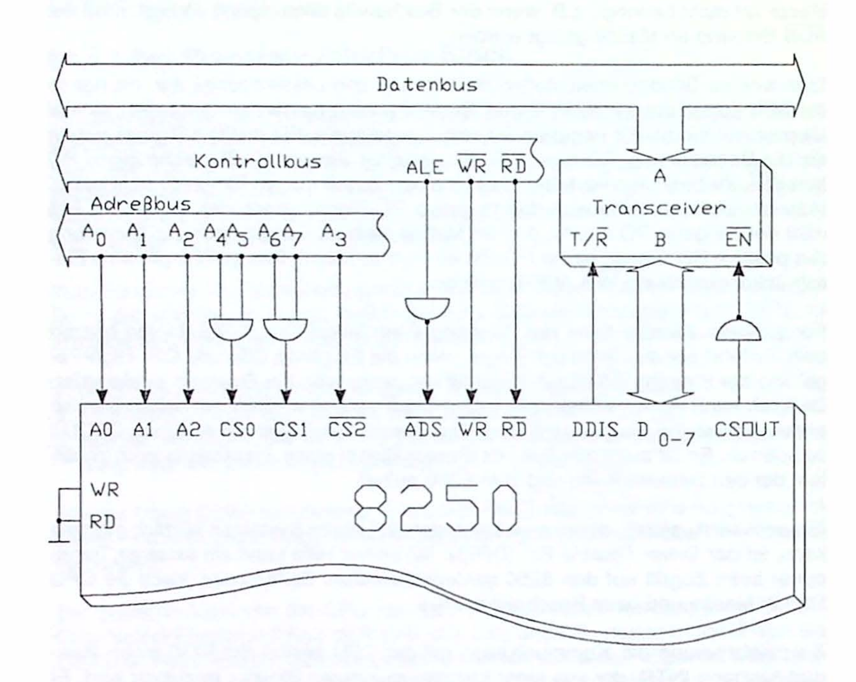


Bild 4-50. Das Prozessor-Interface des 8250

4.6.2.2 Der UART-Block

Die Taktversorgung für den 8250 erfolgt über den internen Oszillator, der an den Anschlüssen XTAL1 und XTAL2 ein frequenzbestimmendes RC-Glied oder einen Schwingquarz benötigt. Die Maximalfrequenz beträgt 10 MHz. Da diese Frequenz im Transmitter-Kontrollblock durch 16 geteilt wird (Bild 4-51), ist die maximale Baudrate 625 KHz. Der Baustein kann aber auch von einer externen Taktquelle versorgt werden. In diesem Fall ist der Pin XTAL1 als Eingang zu verwenden, der Pin XTAL2 bleibt offen.

Über den UART-Block laufen die Daten in serieller Form. Bei der Datensendung werden die Daten in paralleler Form im Sendespeicher (Bild 4-51) abgelegt und durch das nachgeschaltete Schieberegister seriell über den Ausgang Seriell Out (SOUT) ausgegeben (Pin 11). Den dazu erforderlichen Takt erhält das Schieberegister vom programmierbaren Baudraten-Generator. Die übermittelte Wortlänge ist auf 5, 6, 7 oder 8 Datenbits programmierbar. Der Transmitter-Teil fügt ein Start-, ein Paritäts- und ein Stoppbit hinzu. Unter den Stoppbits hat man die Wahl zwischen 1, 1½ oder 2 Stoppbits.

Der Signalzustand 1, der in der seriellen Datenübertragung auch Mark genannt wird, wird durch High-Pegel am Ausgang SOUT repräsentiert, der Signalzustand 0, auch Space genannt, durch Low-Pegel. Im nichtaktiven Zustand führt der Ausgang High-Pegel. Diese Pegel müssen in der V.24- bzw. RS-232-C-Schnittstelle in entsprechende Spannungspegel umgewandelt werden: Es gilt für **Mark** eine Spannung zwischen -3 V und -15 V und für **Space** eine Spannung zwischen +3 V und +15 V.

Der Ausgang $\overline{\text{BAUDOUT}}$ gibt eine Frequenz aus, die das 16fache der gesendeten Datenrate ausmacht. Das $\overline{\text{BAUDOUT}}$-Signal ist somit die Oszillatorfrequenz geteilt durch den in den Registern DLL und DLM voreingestellten Wert. Es sind Teilerwerte von 1 bis 2^{16}-1 für die Ausgabe von RS-232-C-Standard-Baudraten programmierbar, wenn einer der drei industriellen Standard-Baudraten-Quarze (1,8432 MHz; 2,4576 MHz oder 3,072 MHz) verwendet wird. Dieses Signal wird üblicherweise dazu benutzt, um den Datenempfänger zu takten. Es kann aber auch an den Eingang RCLK des 8250 geführt werden und so den Datenempfangsteil mit einer festen Frequenz versorgen.

Der Eingang RCLK verlangt seinerseits die 16fache Taktfrequenz der benutzten Baudrate und taktet nach einer entsprechenden Teilung des Signals das Schieberegister des Empfangsteils. Sollten Sender und Empfänger sich auf eine gemeinsame Baudrate geeinigt haben, kann der Ausgang $\overline{\text{BAUDOUT}}$ mit dem Eingang RCLK verbunden werden.

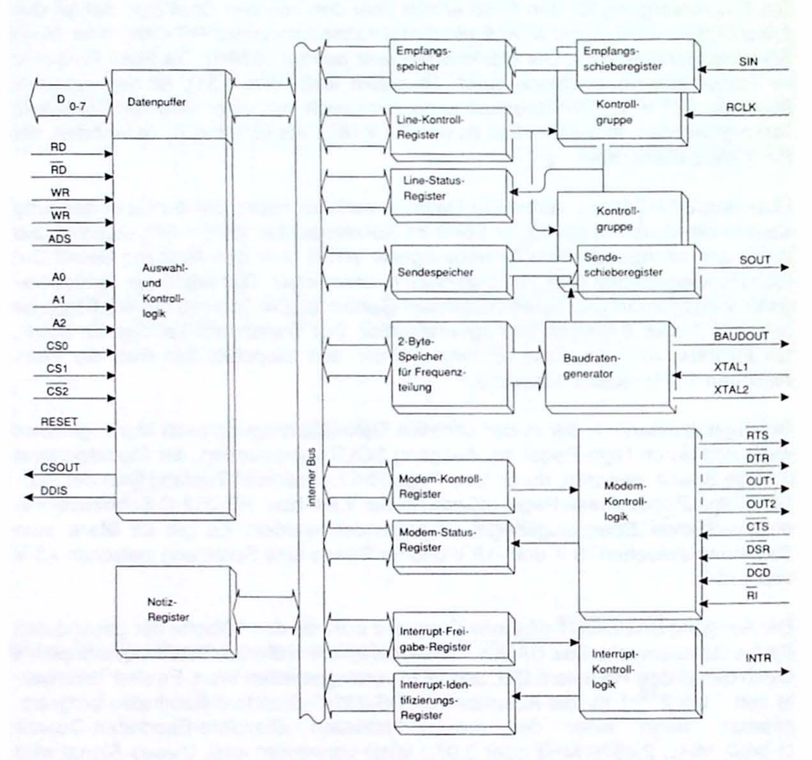


Bild 4-51. Das Blockdiagramm des 8250

Die Daten fließen in serieller Form über den Eingang SIN in das Empfangsschieberegister. Der Pin SIN stellt den seriellen Eingang für die Daten dar, die aus dem Modem oder der Kommunikationsleitung in den 8250 gelangen sollen. Auch hier bedeutet Mark wiederum High-Pegel und Space Masse.

4.6.2.3 Der Modem-Block

Für eine Zusammenarbeit mit einem Modem bietet der Modem-Block die notwendigen Signale und akzeptiert seinerseits Kontrollkommandos vom Modem.

Der Eingang $\overline{CTS}$ (Clear To Send) zeigt dem 8250 mit Massepegel an, daß das Modem bereit ist, Daten über die Datenleitung SOUT zu empfangen. Bei High-Pegel sollten keine Daten gesendet werden, da sie in diesem Fall vom Modem nicht weiterverarbeitet werden und verloren gehen. Der logische Pegel des Eingangs $\overline{CTS}$ findet sich in Bit 4 des Modem-Statusregisters, wo ihn auch die CPU lesen kann. Treten seit dem letzen Lesevorgang durch die CPU Änderungen in der $\overline{CTS}$-Leitung auf, wird das Bit 0 des Modem-Statusregisters gesetzt. Auf Wunsch kann dieses Setzen einen Interrupt erzeugen, der der CPU mitteilt, daß sie mit dem Datentransfer beginnen kann oder ihn stoppen muß.

Eine ähnliche Funktion übt der Eingang $\overline{DSR}$ (Data Set Ready) aus. Über ihn zeigt das Modem dem 8250 an, daß es zum Datenaustausch bereit ist. Im Gegensatz zum $\overline{CTS}$-Signal wird über die Leitung $\overline{DSR}$ die Betriebsbereitschaft des Modems angezeigt. Auch dieser Eingang wird im Modem-Statusregister Bit 5 zwischengespeichert. Eine Änderung an diesem Eingang bewirkt das Setzen von Bit 1, das nach einem Lesevorgang durch die CPU rückgesetzt wird. Auch hierbei kann ein Interrupt ausgelöst werden. Der Massepegel zeigt nur den Zustand des Datenendgeräts an (ob aus- oder eingeschaltet), macht aber keine Aussage, ob eine Verbindung mit einem zweiten Modem am anderen Ende der Telefonleitung zustande gekommen ist.

Der dritte Kontrolleingang $\overline{DCD}$ (Data Carrier Detect) erhält dann seinen Sinn, wenn die Daten beispielsweise über die V.24- oder RS-232-C-Schnittstelle an das Modem gesendet werden. Das Endgerät prüft, ob der Spannungspegel der übertragenen Daten innerhalb des vom Hersteller vorgegeben Bereichs liegt. Ist das der Fall, legt das Modem die Leitung $\overline{DCD}$ an Masse. Treten Störungen auf oder werden keine Daten gesendet, führt die Leitung High-Pegel. Auch dieser Pegel ist im Modem-Kontrollregister Bit 7 nachlesbar. Pegeländerungen an diesem Eingang werden mit einer Eins im Bit 3 abgelegt. Auf Wunsch kann dabei ein Interrupt ausgelöst werden.

Der vierte Eingang ist ein Indikator für das Läuten des Telefons $\overline{\text{RI}}$ (Ring Indikator). Masse zeigt dem 8250 an, daß das Modem über die Postleitung angewählt wird und ein Klingelsignal gerade anliegt. Das Signal ist also nur für den Zeitraum aktiv, in dem das Läutesignal von der Telefonleitung in das Modem kommt. In den Pausen führt die Leitung High-Pegel. Auch hier wird mit Masse am Eingang $\overline{\text{RI}}$ das Bit 6 im Modem-Statusregister gesetzt. Die positive Flanke an diesem Eingang wird mit einer 1 in Bit 2 der CPU angezeigt. Wie in den drei vorangehenden Fällen kann auch hierdurch ein Interrupt erzeugt werden.

In jedem Fall tut die CPU bei einer Interruptanforderung gut daran, zunächst die Inhalte des Modem-Statusregisters zu lesen, um die Herkunft des Interrupts zu erkennen. Dabei werden eventuell vorhandene Einsen in den Bits 0 bis 3 rückgesetzt.

Ein Steuersignal, das der 8250 aussendet, ist das $\overline{\text{RTS}}$-Signal (Request To Send). Dieser Ausgang wird benutzt, um mit Masse dem Datenendgerät anzuzeigen, daß im 8250 Daten bereit zum Senden anstehen. Im Halb-Duplex-Betrieb dient diese Leitung zur Richtungssteuerung des Datenflusses. Der $\overline{\text{RTS}}$-Pin wird an Masse gesetzt, indem man eine Eins in das Bit 1 des Modem-Kontrollregisters schreibt.

Das zweite Steuersignal ist das $\overline{\text{DTR}}$-Signal (Data Terminal Ready). Mit Masse auf dieser Leitung wird dem Modem angezeigt, daß der 8250 bereit zum Datenempfang ist. In einigen Fällen kann der $\overline{\text{DTR}}$-Pin als Anzeige für das Vorhandensein der Betriebsspannung verwendet werden. High-Pegel auf dieser Leitung bewirkt, daß das Modem die Datenübertragung abbricht und die Verbindung zum Gesprächsteilnehmer aufgibt (d.h. den "Hörer" auflegt). Der Pegel dieser Leitung kann durch Beschreiben von Bit 0 des Modem-Kontrollregisters gewählt werden. Es bedeutet das Schreiben einer Eins die Ausgabe von Masse und das Schreiben einer Null die Ausgabe von High-Pegel.

Die Ausgänge $\overline{\text{OUT1}}$ und $\overline{\text{OUT2}}$ sind nichts anderes als programmierbare Portpins, die ohne konkrete Aufgabe zur beliebigen Verwendung benutzt werden können. Sie werden durch Beschreiben der Bits 2 bzw. 3 mit einer Eins an Masse und mit Beschreiben einer Null an High-Pegel gelegt.

4.6.3 Die internen Register des 8250

Der 8250 stellt dem Benutzer drei Typen von internen Registern zur Verfügung:

1. Kontrollregister. Dazu gehören:
   a. Zwei Register von je 8 Bit Breite, die den Frequenzteiler zur Baudraten-Erzeugung enthalten.
   b. Das Line-Kontrollregister, in dem die wichtigsten Daten zur Datensendung, wie Wortlänge, Stoppbit, Paritätsbit etc., enthalten sind.
   c. Das Interrupt-Freigaberegister, das die komplexe Interrupt-Struktur steuert.
   d. Das Modem-Kontrollregister, das die Arbeit des Modem-Blocks überwacht.
2. Statusregister. Dazu gehören:
   a. Das Line-Statusregister
   b. Das Modem-Statusregister
   c. Das Interrupt-Identifizierungsregister
3. Datenregister. Dazu gehören:
   a. Der Sendespeicher
   b. Der Empfangsspeicher
   c. Das Notizregister

Insgesamt verfügt der 8250 also über elf interne Register, auf die man durch die entsprechende Adressierung an den Eingängen A_{0-2} zugreifen kann. Mit diesen drei Adreßeingängen sind allerdings nur acht verschiedene Register adressierbar. Die restlichen Register teilen sich die Adressen mit drei anderen.

Woher nun weiß der Baustein, auf welches Register gerade zugegriffen wird? Der Sendespeicher und der Empfangsspeicher teilen sich eine gemeinsame Adresse. Beim Lesen erhält man den Inhalt des Empfangsspeichers und beim Schreiben gehen die Daten intern gesteuert in den Sendespeicher. Ein Lesen des Sendespeichers ist also genauso unmöglich wie das Beschreiben des Empfangsspeichers. Die beiden Register, deren Inhalte für die Frequenzteilung verwendet werden, teilen sich die Adressen einmal mit dem Sende-/Empfangsspeicher und zum anderen mit dem Interrupt-Freigaberegister.

Das Problem der Adressengleichheit ist auf folgendem Weg gelöst:

Im Inneren des 8250 gibt es ein Flip-Flop, dessen Ausgang einen Daten-Multiplexer steuert. Dieser Multiplexer läßt bei gesetztem Flip-Flop die Daten in die Register für die Frequenzteilung fließen. Bei gelöschtem Flip-Flop steht der Datenbus mit den seriellen Speichern, bzw. dem Interrupt-Freigaberegister in Verbindung. Dieses Flip-Flop selbst ist Bestandteil des Line-Kontrollregisters und wird dort durch das Bit 7 repräsentiert. Da nach einem Reset alle Bits des Line-Kontrollregisters Nullen aufweisen, muß Bit 7 vor dem Beschreiben des Frequenzteilers ausdrücklich gesetzt werden. Nach dem Beschreiben ist es gleich zu löschen, da sonst kein Zugriff auf die seriellen Register möglich ist. Auf die Adressierung der restlichen Register hat dieses Bit keinen Einfluß. Die Adressierung aller Register ist in Tabelle 4-31 zu finden.

A_2	A_1	A_0	**Register**
Bit 7 des Line-Kontrollregisters = 0			
0	0	0	Empfangsspeicher (beim Lesen)
0	0	0	Sendespeicher (beim Schreiben)
0	0	1	Interrupt-Freigaberegister
Bit 7 des Line-Kontrollregisters = 1			
0	0	0	Register für das Low-Byte der Frequenzteilung
0	0	1	Register für das High-Byte der Frequenzteilung
Bit 7 des Line-Kontrollregisters = 0 oder 1			
0	1	0	Interrupt-Identifizierungsregister
0	1	1	Line-Kontrollregister
1	0	0	Modem-Kontrollregister
1	0	1	Line-Statusregister
1	1	0	Modem-Statusregister
1	1	1	Notizregister

Tabelle 4-31. Die Adressierung der internen Register

4.6.3.1 Das Line-Kontrollregister

Das Format des Datencharakters wird durch das Line-Kontrollregister bestimmt. Da der Inhalt des Registers auch gelesen werden kann, entfällt die Notwendigkeit einer separaten Speicherung der eingestellten Werte im Systemspeicher.

Die einzelnen Bits haben folgende Bedeutung:

Bit 0 und Bit 1 bestimmen die Wortlänge , d.h. die Zahl der Bits, die bei der seriellen Übertragung pro Charakter gesendet oder empfangen werden sollen. Dabei gilt folgende Zuordnung:

Bit 1	Bit 0	Wortlänge
0	0	5 Bits
0	1	6 Bits
1	0	7 Bits
1	1	8 Bits

Bit 2 bestimmt die Zahl der Stoppbits in jedem gesendeten Wort. Eine Null erzeugt ein Stoppbit, eine Eins erzeugt bei der Wortlänge von 5 Bits 1,5 Stoppbits bei den restlichen Wortlängen 2 Stoppbits. Der Empfangsteil erwartet ebenfalls zwei Stoppbits.

Bit 3 steuert die Ausgabe oder den Empfang eines Paritätsbits. Ist in diesem Bit eine Eins, wird beim Senden ein Paritätsbit zwischen dem letzen Bit des Datenworts und dem Stoppbit eingefügt bzw. beim Empfang überprüft. Ein Paritätsbit stellt also ein zusätzliches Bit des zu übertragenden Worts dar, mit dessen Hilfe dem Empfänger das Erkennen eines einfachen Fehlers möglich ist. Ein Erkennen doppelter Fehler ist damit nicht möglich.

Bit 4 bestimmt bei gesetztem Bit 3 die Art des erzeugten Paritätsbits. Ist der Inhalt von Bit 4 eine Null, wird der logische Pegel des Paritätsbits so gewählt, daß über die Leitung einschließlich des Paritätsbits selbst immer eine ungerade Zahl von Einsen gesendet wird. Man nennt das eine **ungerade Parität**. Beispiel: Sollen sechs Bits mit Inhalt 101001 gesendet werden, fügt der 8250 als Paritätsbit eine Null an, damit die Gesamtzahl an Einsen ungerade bleibt: 101001+0. Soll das Bitmuster 001010 gesendet werden, wird als Paritätsbit eine Eins angefügt, um die Zahl der Einsen ungerade zu machen: 001010+1.

Weist der Inhalt von Bit 4 eine Eins auf, sorgt die Sendelogik dafür, daß immer eine gerade Zahl von Einsen über die Leitung geht. Man nennt das eine **gerade Parität**. Beispiel: Sollen sechs Bits mit Inhalt 101001 gesendet werden, fügt der 8250 als Paritätsbit eine Eins an, damit die Gesamtzahl an Einsen gerade wird: 101001+1. Soll das Bitmuster 001010 gesendet werden, wird als Paritätsbit eine Null angefügt, um die Zahl der Einsen gerade zu lassen: 001010+0.

Beim Datenempfang wird in ähnlicher Weise auf eine gerade oder ungerade Zahl von Einsen geachtet.

Bit 5: Bei eingeschalteter Paritätsprüfung (Bit 3 = 1) bewirkt eine Eins in Bit 5, daß beim Senden bzw. beim Empfang der logische Pegel des Paritätsbits invertiert wird. Das gestattet dem Anwender, die Parität in einen bekannten Zustand zu versetzen und für den Empfänger das Paritätsbit mit einem bekannten Zustand zu vergleichen. Hier ist nun das Paritätsbit genau dann eine Eins, wenn das Datenwort bei gerader Parität eine gerade Zahl von Einsen, und bei ungerader Parität eine ungerade Zahl von Einsen aufweist.

Bit 6 übt eine Kontrolle über den Pin SOUT aus, beeinflußt aber nicht die Sende-Logik. Wird dieses Bit mit einer Eins beschrieben, wird der Ausgang SOUT unmittelbar an Masse gelegt und bleibt dort solange, bis Bit 6 gelöscht wird. Mit dieser Abschaltmöglichkeit ist die CPU in der Lage, ein Datenendgerät im Kommunikationssystem des Computers zu überwachen. Damit beim Abschalten des Ausgangs keine fehlerhaften oder unerwünschte Daten gesendet werden, sollten beim Setzen des Bits folgende Punkte beachtet werden:

1. Man warte bis der Sendespeicher leer ist. Das ist an einer Eins in Bit 5 des Line-Statusregisters erkennbar. Danach schreibt man einen Block von Nullen in den Sendespeicher.
2. Man warte erneut, bis der Sendespeicher leer ist, und setze nun das Unterbrechungsbit 6.
3. Eine Freigabe des Ausgangs SOUT sollte nicht eher erfolgen, bevor auch das Schieberegister leer ist. Das ist an einer Eins in Bit 6 des Line-Statusregisters erkennbar. Wenn die normale Sendearbeit wieder aufgenommen werden soll, ist Bit 6 zu löschen.

Bit 7 hat keinen direkten Einfluß auf die Verarbeitung serieller Daten. Es ist vielmehr verantwortlich für den parallelen Datenstrom, der von der CPU kommt oder zu ihr fließt. Eine Eins lenkt die Daten in die Register, die für die Frequenzteilung im UART-Block zuständig sind, eine Null verbindet den Datenbus mit dem Sende-/Empfangsspeicher oder dem Interrupt-Freigaberegister.

4.6.3.2 Das Line-Statusregister

Das Line-Statusregister versorgt die CPU mit Informationen über den augenblicklichen Zustand der Datenübertragung. Daher ist es das erste Register, das von der CPU gelesen wird, wenn ein Interrupt auftritt oder gewisse Informationen vom 8250 gewünscht werden.

Für die Empfangseinheit stehen drei Bits zur Fehlermeldung zur Verfügung. Es wird damit ein Überlauf, ein Paritätsfehler oder ein Frame-Fehler dokumentiert. Unter Frame versteht man die Gesamtheit aller Bits, die pro Wort über die serielle Datenleitung laufen, also Start-, Daten-, Paritäts- und Stoppbits. Beim Eingang der Stoppbits werden die Fehlerbits gesetzt, sofern ein solcher erkannt wurde. Diese **Fehlerbits** werden bei einem nachfolgenden fehlerfreiem Wort nicht gelöscht. Sie reflektieren aber nur dann den Zustand des letzten Worts, wenn kein Überlauf auftrat. Mit dem **Überlaufbit** wird angezeigt, daß ein Wort im Empfangsspeicher von einem neuen überschrieben wurde, bevor das alte von der CPU gelesen wurde. Das alte Wort ist somit verloren. Das **Frame-Bit** gibt Auskunft darüber, ob das eingegangene Wort eine unkorrekte Zahl von Stoppbits aufweist. Ein Fehler dieser Art kann eintreten, wenn das erforderliche Stoppbit fehlt oder wenn es zu kurz ist, um erkannt zu werden. Ein **Paritätsfehler** wird dann gemeldet, wenn die programmierte Parität vom Sender mit der errechneten Parität des Empfängers nicht übereinstimmt. Es brauchen zu diesem Zweck nur die beiden Paritätsbits verglichen zu werden.

Das **Datenübertragungsabbruchbit** zeigt den Eingang des Abbruchworts an. Das Abbruchwort ist ein ungültiger Datencharakter, in dem alle Bits einschließlich Paritäts- und Stoppbit Nullen sind.

Beim Lesen des Line-Statusregisters werden die Bits 1 bis einschließlich 4 gelöscht.

Die Bits des Line-Statusregisters haben im einzelnen folgende Bedeutung:

Bit 0 zeigt mit einer Eins an, daß das seriell eingegangene Wort aus dem Schieberegister in den Empfangsspeicher kopiert wurde und dort zum Lesen durch die CPU bereitsteht. Das Rücksetzen des Bits erfolgt automatisch durch das Lesen des Empfangsspeichers.

Bit 1 zeigt mit einer Eins an, daß ein Datenwort im Empfangsspeicher überschrieben wurde, ohne daß es von der CPU gelesen worden war. Es ist das Bit, mit dem ein Überlauffehler gekennzeichnet wird. Die alte Information ist für die CPU damit verloren.

Bit 2 meldet mit einer Eins einen aufgetretenen Paritätsfehler. Das eingegangene Bitmuster hat in diesem Fall keine gerade (ungerade) Zahl von Einsen und stimmt somit nicht mit der programmierten geraden (ungeraden) Parität überein.

Bit 3 wird bei einem Frame-Fehler auf Eins gesetzt. Ein Frame-Fehler tritt dann auf, wenn das eingehende Bitmuster über kein gültiges Stoppbit verfügt, d.h. wenn nach dem letzten Datenbit oder dem angehängten Paritätsbit Massepegel festgestellt wird. Denn Stoppbits besitzen immer High-Pegel.

Bit 4 wird auf High-Pegel gesetzt, wenn am Eingang SIN ausschließlich Nullen (Start-, Daten-, Paritäts- und Stoppbits) eingehen. Sind alle Bits bis auf das Stoppbit Nullen, ist das ein gültiges Datenwort und Bit 4 wird nicht gesetzt.

Bit 5 meldet mit einer Eins, daß der Sendespeicher leer ist und die CPU neue Sendedaten in ihn schreiben kann. Dieses Bit wird gesetzt, wenn der 8250 die Daten aus dem Sendespeicher in das Sendeschieberegister kopiert hat und nicht erst, wenn der Sendevorgang abgeschlossen ist. Es wird automatisch zurückgesetzt, wenn die CPU den Sendespeicher neu beschreibt. Auf Wunsch kann beim Setzen des Bits ein Interrupt der Priorität 3 erzeugt werden.

Bit 6 wird dann auf Eins gesetzt, wenn sowohl der Sendespeicher als auch das Sendeschieberegister leer sind. Beim Laden eines Bytes in den Sendespeicher wird die Eins in diesem Bit rückgesetzt.

Bit 7 erscheint beim Lesen des Line-Statusregisters immer als eine Null.

Die Bits 1 bis 4 melden das Auftreten eines Fehlers beim seriellen Datenempfang. Sie können nach Freigabe einen Interrupt der Priorität 1 erzeugen. Dazu muß das Bit 2 im Interrupt-Freigaberegister gesetzt sein.

4.6.3.3 Das Modem-Kontrollregister

Das Modem-Kontrollregister kontrolliert die Verbindung mit einem Modem oder Datenendgerät. Das Register kann sowohl gelesen als auch beschrieben werden. Man beachte, daß die Ausgänge $\overline{RTS}$, $\overline{DTR}$, $\overline{OUT1}$ und $\overline{OUT2}$ direkt die logischen Zustände einiger Bits in diesem Register wiederspiegeln. **Wichtig: Ein gesetztes Bit bewirkt Low-Pegel am Ausgang, ein gelöschtes Bit bewirkt High-Pegel.**

Die Bits des Modem-Kontrollregisters haben folgende Bedeutung:

Bit 0 bestimmt den logischen Zustand des Ausgangs $\overline{DTR}$ (Data Terminal Ready). Ein gesetztes Bit bewirkt Masse am Ausgang, ein gelöschtes Bit High-Pegel.

Bit 1 kontrolliert den Pegel am Ausgang $\overline{RTS}$ (Request to Send). Ein gesetztes Bit bewirkt Masse am Ausgang, ein gelöschtes Bit High-Pegel.

Bit 2 bestimmt den logischen Pegel am Ausgang $\overline{OUT1}$. Ein gesetztes Bit bewirkt Masse am Ausgang, ein gelöschtes Bit High-Pegel.

Bit 3 bestimmt den logischen Pegel am Ausgang $\overline{OUT2}$. Ein gesetztes Bit bewirkt Masse am Ausgang, ein gelöschtes Bit High-Pegel.

Bit 4 gestattet das diagnostische Testen des 8250 mittels einer lokalen Rückkopplungsschleife. Wenn dieses Bit auf Eins gesetzt wird, wird der serielle Ausgang SOUT auf High-Pegel gezogen und gleichzeitig der serielle Eingang abgeschaltet, d.h. vom Empfangsschieberegister getrennt. Der Ausgang des Sendeschieberegisters wird in einer internen Schleife mit dem Eingang des Empfangsschieberegisters verbunden. Die vier Modem-Kontrolleingänge $\overline{CTS}$, $\overline{DSR}$, $\overline{DCD}$ und $\overline{RI}$ werden abgeschaltet. Die vier Modem-Signale $\overline{DTR}$, $\overline{RTS}$, $\overline{OUT1}$ und $\overline{OUT2}$ werden intern mit den vier Modem-Eingängen verbunden und die entsprechenden Modem-Ausgänge führen dabei High-Pegel.
Bei diesem Testmodus werden also die Daten, die gesendet werden, sofort wieder empfangen. Man kann somit nicht nur den Baustein 8250 auf korrekte Arbeitsweise testen, sondern auch die richtige Wirkung der in den verschiedensten Registern gesetzten Bits und die entsprechenden Interrupt-Routinen im Programmspeicher der CPU überprüfen. Die interne Logik, insbesondere die der Interrupt-Steuerung ist voll funktionstüchtig. Allerdings erzeugen jetzt die vier unteren Bits des Modem-Kontrollregisters anstelle der Modem-Eingänge einen Interrupt.

Bit 4 bis Bit 7: Diese Bits erscheinen beim Lesen immer als Nullen. Ein Schreiben beliebiger Werte in diese Bits hat keinen Einfluß auf den Inhalt des Registers.

4.6.3.4 Das Modem-Statusregister

Im Modem-Statusregister findet die CPU die logischen Pegel, die das Modem oder ein anderes Datenendgerät an die Kontrolleingänge des Modem-Blocks legt. Dadurch kann die CPU über den System-Datenbus die Modem-Signale lesen. Zusätzlich zu diesen aktuellen Eingangspegeln findet die CPU in vier weiteren Sensorbits Informationen darüber, ob sich der logische Pegel einer Signalleitung seit dem letzten Lesevorgang geändert hat. Ihr Inhalt wird automatisch beim Lesen durch die CPU rückgesetzt.

Die Signalleitungen, die in den Modem-Block führen, sind $\overline{CTS}$ (Pin 36), $\overline{DSR}$ (Pin 37), $\overline{RI}$ (Pin 39) und $\overline{DCD}$ (Pin 38). Die Bits 4 bis 7 repräsentieren den logischen Pegel dieser Eingänge. Man beachte auch hier, daß die logischen Pegel im Modem-Statusregister invertiert abgelegt werden. High-Pegel am Eingang erzeugt Low-Pegel im entsprechenden Bit, Masse am Eingang setzt das Bit auf 1. Wenn der Interrupt für den Modem-Status im Interrupt-Freigaberegister (Bit 3) freigegeben ist, führt eine Änderung eines logischen Pegels an einem Eingang nicht nur zum Setzen des entsprechenden Sensorbits (Bit 0 bis Bit 3), sondern löst einen Interrupt der Priorität 4 aus.

Die Bedeutung der Bits des Modem-Statusregisters im einzelnen:

Bit 0: Dieses Sensorbit zeigt mit einer Eins an, daß sich der Zustand des Eingangs $\overline{CTS}$ seit dem letzten Lesen durch die CPU geändert hat.

Bit 1: Dieses Sensorbit zeigt mit einer Eins an, daß sich der Zustand des Eingangs $\overline{DSR}$ seit dem letzten Lesen durch die CPU geändert hat.

Bit 2: Dieses Sensorbit zeigt mit einer Eins an, daß das Klingelsignal der Post im Modem zu Ende ist. Es wird genau dann gesetzt, wenn der Pegel am Eingang $\overline{RI}$ von Masse nach Plus wechselt. Bei dem Pegelwechsel von Plus nach Masse wird es nicht gesetzt.

Bit 3: Dieses Sensorbit zeigt mit einer Eins an, daß sich der Zustand des Eingangs $\overline{DCD}$ seit dem letzten Lesen durch die CPU geändert hat.

Bit 4 spiegelt in invertierter Form den logischen Zustand des Eingangs Clear-to-Send ($\overline{\text{CTS}}$) wieder. Eine Eins bedeutet für die CPU, daß das Modem bereit ist, Daten vom Sendeausgang SOUT des 8250 aufzunehmen. Wenn sich der 8250 im Testmodus befindet, ist der Inhalt des Bits identisch mit dem Inhalt des Bits 1 (RTS) im Modem-Kontrollregister.

Bit 5 spiegelt in invertierter Form den logischen Zustand des Eingangs Data Set Ready ($\overline{\text{DSR}}$) wieder. Eine Eins bedeutet für die CPU, daß das Modem oder das Datenendgerät betriebsbereit ist. Damit Daten gesendet werden können, müssen Bit 5 und Bit 4 eine Eins aufweisen. Wenn sich der 8250 im Testmodus befindet, ist der Inhalt des Bits identisch mit dem Inhalt des Bits 0 (DTR) im Modem-Kontrollregister.

Bit 6 spiegelt in invertierter Form den logischen Zustand des Eingangs Ring Indikator ($\overline{\text{RI}}$) wieder. Eine Eins bedeutet für die CPU, daß das Postnetz im Modem einen Rufton erzeugt. Wenn sich der 8250 im Testmodus befindet, ist der Inhalt des Bits identisch mit dem Inhalt des Bits 2 (OUT1) im Modem-Kontrollregister.

Bit 7 spiegelt in invertierter Form den logischen Zustand des Eingangs Data Carrier Detect ($\overline{\text{DCD}}$) wieder. Eine Eins bedeutet für die CPU, daß die gesendeten Daten mit den richtigen Pegeln (z.B. $\pm$ 12 V für die RS-232-C-Schnittstelle) am Modem ankommen. Wenn sich der 8250 im Testmodus befindet, ist der Inhalt des Bits identisch mit dem Inhalt des Bits 3 (OUT2) im Modem-Kontrollregister.

Bits 4 bis 7 spiegeln die Eingänge des Modem-Blocks wieder. Beim Lesen des Registers werden lediglich die Sensorbits rückgesetzt, die Statusbits werden dadurch nicht geändert. Tritt eine Änderung an irgend einem Eingang auf und ist das betreffende Sensorbit noch nicht rückgesetzt, wird diese Änderung nicht angezeigt, d.h. das Sensorbit bleibt weiter gesetzt. Tritt jedoch eine Änderung beim Lesen durch die CPU auf, wird das Bit unmittelbar nach dem Lesen wieder gesetzt.

4.6.3.5 Die Register für die Auswahl der Baudraten

Der 8250 besitzt einen programmierbaren Baudraten-Generator, der den Takt durch jeden gewünschten Wert teilt. Als Teiler sind die Zahlen von 1 bis 2^{16}-1 möglich. Die ausgegebene Frequenz des Baudraten-Generators ist das 16fache der Datenrate [Teiler n = Oszillatorfrequenz:(Baudrate • 16)]. Der 16-Bit-Teiler n wird in zwei 8-Bit-Register abgelegt. Diese Register sind bei der Initialisierung mit dem gewünschten Wert beschrieben worden, können aber auch jederzeit geändert werden.

Beispiele für die Berechnung des Teilers n:

Gegeben:	Oszillatorfrequenz von 1,8432 MHz
Gewünscht:	Eine Baudrate von 1200
Rechnung:	Teiler n = Oszillatorfrequenz:(Baudrate • 16)
	Teiler n = 1843200:(1200 • 16)
Ergebnis:	Teiler n = 96_d = 60_h
Probe:	Der Teiler 60_h teilt die Eingangsfrequenz von 1,8432 MHz auf einen Wert von 19200 Hz, der das 16fache der gewünschten Baudrate darstellt.

Das Low-Byte des Registers für die Frequenzteilung ist somit mit 0110 0000, das High-Byte mit dem Wert 0000 0000 zu beschrieben.

4.6.3.6 Der Empfangsspeicher

Der Empfangsspeicher kann für die Aufnahme von fünf, sechs, sieben oder acht Datenbits programmiert werden. Für Wörter, die weniger als acht Bits besitzen, erfolgt die Eintragung der Bits rechtsbündig, d.h. das Datenbit 0 findet sich in Bit 0 des Empfangsspeichers. Beim Empfang ist Bit 0 das erste Bit, das aufgenommen wird. Beim Lesen des Empfangsspeichers durch die CPU erscheinen die nicht benutzten Bits als Nullen.

Die Daten, die am Pin SIN eingehen, werden in das Empfangsschieberegister mit dem sechzehnten Teil des Taktes geschoben, der am Eingang RCLK anliegt. Dieser Takt wird auf der Grundlage des eingehenden Startbits mit den eingehenden Daten synchronisiert. Sobald sich das vollständige Wort im Empfangsschieberegister befindet, wird es parallel in den Empfangsspeicher kopiert und Bit 0 im Line-Statusregister gesetzt.

Diese Doppelspeicherung der eingehenden Daten gestattet eine kontinuierliche Aufnahme von Daten, ohne daß ein eingegangenes Wort verloren geht. Während das Schieberegister mit der Aufnahme des zweiten Worts beschäftigt ist, stellt der Empfangsspeicher das erste zum Lesen für die CPU bereit. Wird der Empfangsspeicher vor der vollständigen Aufnahme des zweiten Worts nicht gelesen, ist das erste Wort verloren. In diesem Fall wird das Überlaufbit (Bit 1 im Line-Statusregister) gesetzt.

4.6.3.7 Der Sendespeicher

Der Sendespeicher nimmt die Daten vom System-Datenbus auf und speichert sie solange, bis das Sendeschieberegister leer und für die Aufnahme neuer Daten bereit ist. Die Wortlänge für den Sender und den Empfänger, sowie die Zahl der Stoppbits ist immer gleich groß. Ist die Wortlänge geringer als acht Bits, werden die unbenutzten Bits beim Beschreiben des Sendespeichers einfach ignoriert.

Das Datenbit 0 ist das erste seriell übertragene Datenbit. Der Zustand des Sendespeichers wird in Bit 5 des Line-Statusregisters wiedergegeben. Bit 6 des Line-Statusregisters zeugt an, daß sowohl Sendespeicher als auch Sendeschieberegister leer sind.

4.6.3.8 Das Notizregister

Dieses 8-Bit-Schreib-/Leseregister übt keine Steuerungs- oder Kontrollfunktion auf den 8250 aus. Es ist gedacht als ein Register, das als temporärer Speicher für allgemeindienliche Zwecke zur Verfügung steht. Es ist ein 8-Bit-RAM-Platz im 8250.

4.6.3.9 Das Interrupt-Identifizierungsregister

Um die Zusammenarbeit mit der CPU zu erleichtern, stellt der 8250 ein Interrupt-Signal zur Verfügung, das aus den verschiedensten Gründen intern ausgelöst werden kann. Da mehrere Ursachen gleichzeitig auftreten können, besitzt der 8250 vier Interrupt-Ebenen, die mit fallender Priorität nachfolgend aufgeführt sind:

1. Status der Eingangsleitungen (Priorität 1)
2. Empfangsdaten zum Lesen bereit (Priorität 2)
3. Sendespeicher leer (Priorität 3)
4. Status des Modem-Blocks (Priorität 4)

Informationen über die Art des Interrupts sind im Interrupt-Identifizierungsregister zu finden. Solange ein angeforderter Interrupt durch die CPU nicht bedient ist, werden vom 8250 keine weiteren Anforderungen der Hardware anerkannt. Die Löschung des Interrupt-Signals erfolgt auf verschiedenste Weisen, die von der Priorität des Interrupts abhängen (Tabelle 4-32). Das Interrupt-Identifizierungsregister kann nicht beschrieben werden.

Bit 0: Die CPU erkennt an Bit 0, ob eine Interrupt-Anforderung vorliegt oder nicht. Ist der Inhalt des Bits eine 0, liegt eine Interrupt-Anforderung vor und die Bits 1 und 2 können für die CPU als Zeiger für eine entsprechende Interrupt-Routine im Programmspeicher dienen, ist der Inhalt eine 1, liegt keine Interrupt-Anforderung vor. Der Hauptzweck von Bit 0 ist es, Prozessoren, die die Abfragemethode verwenden, ein einfaches Hilfsmittel zur Verfügung zu stellen, mit dessen Hilfe sie schnell eine Interrupt-Anforderung erkennen können.

Bit 1 und Bit 2 beinhalten in binärkodierter Form die Nummer des anstehenden Interrupts. Die Bedeutung der vier möglichen Kombinationen ist in Tabelle 4-32 aufgeführt.

Bit 3 bis Bit 7 haben keine Bedeutung; sie erscheinen beim Lesen als Nullen.

Bit2	Bit 1	Priorität	Interrupt-Herkunft	Register	Rücksetzen des Interrupts
1	1	1	Überlauf, Paritätsfehler, Frame-Fehler oder Übertragungsabbruch	Line-Statusregister Bits 1, 2, 3 oder 4	Lesen des Line-Statusregisters
1	0	2	Empfangsspeicher zum Lesen bereit	Line-Statusregister Bit 0	Lesen des Empfangsspeichers
0	1	3	Sendespeicher leer	Line-Statusregister Bit 5	Lesen des Interrupt-Identifizierungsregisters oder Beschreiben des Sendespeichers
0	0	4	Signale der Leitungen: $\overline{CTS}$, $\overline{DSR}$, $\overline{RI}$ oder $\overline{DCD}$	Modem-Statusregister Bits 0, 1, 2 oder 3	Lesen des Modem-Statusregisters

Tabelle 4-32. Der Inhalt des Interrupt-Identifizierungsregisters

4.6.3.10 Das Interrupt-Freigaberegister

Das Interrupt-Freigaberegister ist ein Schreibregister, mit dessen Hilfe man unabhängig voneinander die Interrupts der verschiedenen Prioritätsebenen freigeben oder sperren kann. Von den acht Bits des Registers werden nur die unteren vier benutzt. Um einen Interrupt einer Herkunftsquelle freizugeben, muß das entsprechende Bit gesetzt werden, zum Sperren muß man es löschen. Durch ein Beschreiben des Interrupt-Freigaberegisters werden die Inhalte der restlichen Register, insbesondere die der Statusregister, nicht beeinflußt.

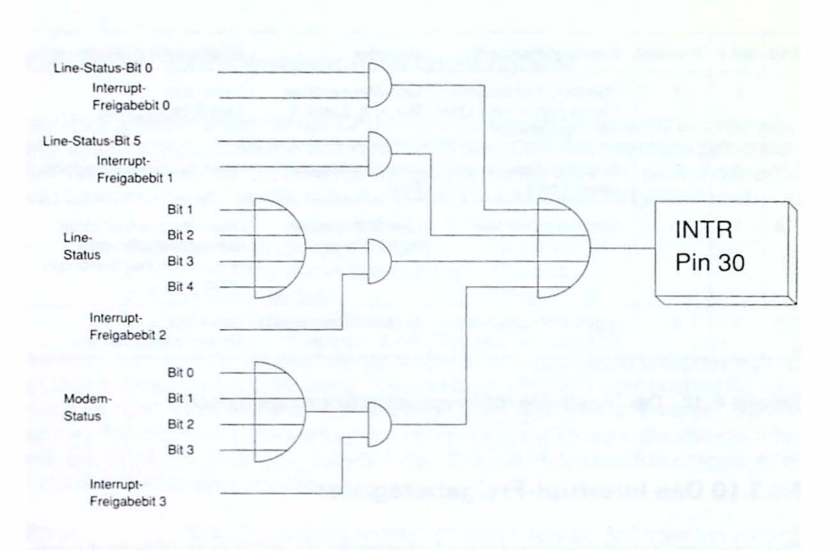


Bild 4-52. Interne Logik zur Erzeugung eines Interrupts

Bit 0 ermöglicht mit Inhalt 1 einen Interrupt, wenn das empfangene Wort zum Lesen für die CPU bereitsteht. Es wird der Interrupt mit der Priorität 2 freigeben.

Bit 1 steuert die Interrupt-Auslösung bei leerem Sendespeicher. Mit einer Eins wird ein Interrupt der Priorität 3 freigegeben.

Bit 2 gibt den Interrupt der Priorität 1 frei, wenn es mit einer Eins beschrieben ist. Eine Eins in den Bits 1 bis 4 des Line-Statusregisters wird in diesem Fall zur Erzeugung eines Interrupts führen.

Bit 3 erlaubt den Interrupt der Priorität 4, wenn eines der Sensorbits des Modem-Statusregisters gesetzt wird.

Bit 4 bis **Bit 7** sind unbenutzt und erscheinen beim Lesen immer als Null.

4.6.4 Die serielle Datensendung

Die serielle Sendeeinheit besteht aus einem Sendespeicher, einem Sendeschieberegister und einer Kontrollogik. Mit Hilfe zweier Bits im Line-Statusregister wird der leere Zustand des Speichers und des Schieberegister angezeigt. Um ein 5- bis 8-Bit-Wort zu senden, muß das Wort über den Datenbus D_{0-7} in den Sendespeicher geschrieben werden. Die CPU sollte nur dann den Sendespeicher beschreiben, wenn Bit 5 im Line-Statusregister eine Eins aufweist und damit dokumentiert, daß der Sendespeicher leer ist. Dieses Bit wird gesetzt, wenn der Inhalt des Speichers in das Schieberegister kopiert und das Startbit gesendet wird.

Wenn die Datensendung abgeschlossen ist und der Sendeteil nicht benutzt wird, weisen die Inhalte der Bits eine Eins auf. Mit dem ersten Wort, das geschrieben wird, gehen beide Bits an Masse und der Sendevorgang wird gestartet. Bit 5 nimmt wieder High-Pegel ein, da das geschriebene Wort augenblicklich in das Schieberegister kopiert wurde, so daß der Sendespeicher für die Aufnahme eines zweiten Worts bereitsteht. Nach dem Schreiben des zweiten Worts geht Bit 5 an Masse und bleibt solange dort, bis das Schieberegister leer ist und das zweite Wort aus dem Sendespeicher kopiert hat. Sobald das letzte Wort gesendet wurde, nehmen beide Bits wieder High-Pegel ein.

4.6.5 Der serielle Datenempfang

Serielle, asynchrone Daten werden über den Eingang SIN in den 8250 eingelesen. Durch einen internen Pull-Up führt der Eingang SIN im nicht aktiven Zustand High-Pegel, und ein Überwachungskreis wartet beständig, bis am Eingang SIN eine negative Flanke auftritt. Wenn eine solche Flanke registriert wird, wird nachfolgend getestet, ob es sich nicht etwa nur um ein Störsignal handelte. Zu diesem Zweck wird ein Zähler rückgesetzt, der mit dem 16fachen der Datenübertragungsrate getaktet wird. Beim Zählerwechsel von 7 nach 8, also in der Mitte des Startbits wird nochmals der Eingang SIN auf Low-Pegel überprüft. Führt er Masse, erkennt der Empfangskreis ein gültiges Startbit und beginnt gemäß der Programmierung im Line-Kontrollregister mit der Datenaufnahme. Führt der Eingang SIN in der Mitte des Startbits wieder High-Pegel, wird der Datenempfang abgebrochen. Somit ist die Empfangseinheit vor einem versehentlichen Start weitestgehend geschützt.

Im Line-Kontrollregister wird die Zahl der Datenbits eines Worts, der Gebrauch eines Paritätsbits und seiner Polarität und die Zahl der Stoppbits festgelegt. Stoppbits sind für den 8250 nichts anderes als die Zeit vor dem nächsten Startbit, in der der Eingang SIN High-Pegel führt. Wenn das Empfangsschieberegister voll ist, wird sein Inhalt in den Empfangsspeicher kopiert, von wo ihn die CPU lesen kann. Dabei wird im Line-Statusregister Bit 0 auf 1 gesetzt, wobei ein Interrupt ausgelöst werden kann. Das Lesen der gesendeten Information durch die CPU löscht die 1 in Bit 0. Erfolgt das Lesen zu spät, ist der ursprüngliche Inhalt des Empfangsspeichers verloren. Um diesen Fehler der CPU anzuzeigen, wird im Line-Statusregister Bit 1 gesetzt. Stellt die Empfangseinheit eine Parität des eingehenden Worts fest, die anders als die programmierte ist, wird durch Setzen von Bit 2 im Line-Statusregister ein Paritätsfehler angezeigt. Des weiteren wird überprüft, ob der Eingang SIN vor dem nächsten Startbit die geforderte Mindestzeit (= Stoppbit) an Plus liegt. Ist diese Zeit unterschritten, wird ein Frame-Fehler durch Setzen des Bits 3 im Line-Statusregister signalisiert.

Um dem Empfangskreis zu signalisieren, daß der Sender die Datenübertragung beenden will, wird über eine Zeit, die länger als der Frame sein muß, Masse am Eingang SIN gelegt. Sobald dies der Empfangskreis erkennt, wird das Bit 4 im Line-Statusregister gesetzt, das über einen Interrupt der CPU den Übertragungsabbruch anzeigen kann.

4.6.6 Der Baudraten-Generator

Der Baudraten-Generator stellt den Takt für den UART-Block. Er ist in der Lage, bei sorgfältiger Wahl des Schwingquarzes und bei entsprechender Teilung Übertragungsraten nach dem ANSI/CCITT-Standard zu erzeugen. Die Taktquelle kann entweder der integrierte Oszillator versehen mit einem externen Kristall an den Anschlüssen XTAL1 und XTAL2 sein oder ein externes Signal, das dem Eingang XTAL1 zugeführt werden muß. In beiden Fällen steht an dem Ausgang $\overline{\text{BAUDOUT}}$ die 16fache Frequenz der gewählten Baudrate zur Versorgung von anderen Peripheriegeräten zur Verfügung. Wenn zwei 8250er in demselben System verwendet werden, kann der zweite mit dem Ausgang $\overline{\text{BOUDOUT}}$ des ersten getaktet werden, allerdings darf dann im zweiten keine Frequenzteilung mehr erfolgen.

Baudrate	Teiler	Abweichung in %
50	2304	0
75	1536	0
110	1047	0,026
134,5	857	0,058
150	768	0
300	384	0
600	192	0
1200	96	0
1800	64	0
2000	58	0,69
2400	48	0
3600	32	0
4800	24	0
7200	16	0
9600	12	0
19200	6	0
38400	3	0
56000	2	2,86

Tabelle 4-33. Baudraten mit einem 1,8432-MHz-Kristall

Die Übertragungsrate ist festgelegt durch die Frequenz am Eingang XTAL1 und durch den Inhalt der beiden Register für die Frequenzteilung. Die durch die eingestellten Werte geteilte Frequenz steht am Ausgang $\overline{\text{BAUDOUT}}$ zu Verfügung und wird intern der Sende- bzw. Empfangseinheit zugeführt, wo sie vor der Verwendung nochmals durch 16 geteilt wird. Beschreibt man die beiden Register für die Frequenzteilung mit Nullen, erfolgt eine Teilung durch Eins, und man hat als Baudrate den sechzehnten Teil der Eingangsfrequenz. Aus der maximalen Oszillatorfrequenz von 10 MHz folgt die maximale Baudrate von 625 kHz.

Baudrate	Teiler	Abweichung in %
50	3072	0
75	2048	0
110	1396	0,026
134,5	1142	0,0007
150	1024	0
300	512	0
600	256	0
1200	128	0
1800	85	0,392
2000	77	0,26
2400	64	0
3600	43	0,775
4800	32	0
7200	21	1,587
9600	16	0
19200	8	0
38400	4	0

Tabelle 4-34. Baudraten mit einem 2,4576-MHz-Kristall

Baudrate	Teiler	Abweichung in %
50	3840	0
75	2560	0
110	1745	0,026
134,5	1428	0,034
150	1280	0
300	640	0
600	320	0
1200	160	0
1800	107	0,312
2000	96	0
2400	80	0
3600	53	0,628
4800	40	0
7200	27	1,23
9600	20	0
19200	10	0
38400	5	0

Tabelle 4-35. Baudraten mit einem 3,072-MHz-Kristall

Für die Erzeugung von Standard-Baudraten empfiehlt sich allerdings die Verwendung von drei marktüblichen Baudraten-Quarzen mit 1,8432 MHz, 2,4576 MHz und 3,072 MHz. Die Tabellen 4-33 bis 4-35 geben Hilfen bei der Programmierung der Register für die Frequenzteilung.

4.6.7 Reset-Operation des 8250

Nach dem Anlegen der Betriebsspannung sollte der Eingang RESET für eine Zeit von mindestens 500 ns an High-Pegel gelegt werden. High-Pegel am Eingang RESET bewirkt folgendes:

1. Die Taktzähler der Sende- und Empfangseinheit werden auf Null gesetzt.

2. Das Line-Statusregister wird bis auf die Bits 5 und 6 gelöscht. Die Bits 5 und 6 werden gesetzt und zeigen dadurch an, daß der Sendespeicher und das Schieberegister leer sind. Des weiteren wird das Modem-Kontrollregister gelöscht. Alle diskreten Verbindungen, Speicherelemente und die verschiedene Logik, die damit verbunden ist, wird gelöscht bzw. ausgeschaltet. Die Inhalte der Register für die Frequenzteilung, der Empfangsspeicher und der Sendespeicher bleiben unverändert.

Zu beachten ist, daß bei nachfolgender Interrupt-Freigabe wegen der gesetzten Bits im Line-Statusregister unmittelbar eine Interrupt-Anforderung bei der CPU erscheint. Tabelle 4-36 faßt die Wirkung eines Resets summarisch zusammen.

Register oder Signal	Wirkung
Interrupt-Freigaberegister	Alle Bits Low-Pegel
Interrupt-Identifizierungsregister	Bit 0 High-, Bits 1 - 7 Low-Pegel
Line-Kontrollregister	Alle Bits Low-Pegel
Modem-Kontrollregister	Alle Bits Low-Pegel
Line-Statusregister	Bit 0 - 4 und Bit 7 Low-, Bits 5,6 High-Pegel
Modem-Statusregister	Bits 0 - 3 Low-Pegel, Bits 4 - 7 Eingangssignale
SOUT	High-Pegel
INTR	Low-Pegel
RTS	High-Pegel
DTR	High-Pegel
OUT1	High-Pegel
OUT2	High-Pegel

Tabelle 4-35. Die Wirkungen des Resets im 8250

4.6.8 Die Programmierung des 8250

Um den Baustein 8250 zu programmieren, sind das Line-Kontrollregister, das Interrupt-Freigaberegister, die beiden Register für die Frequenzteilung und das Modem-Kontrollregister zu beschreiben. Dadurch wird die Wortlänge, Zahl der Stoppbits, Parität, Baudrate und das Modem-Interface festgelegt. Während die Kontrollregister in beliebiger Reihenfolge beschrieben werden können, sollte man das Interrupt-Freigaberegister als letztes beschreiben. Auch während des Betriebs können die Kontrollregister jederzeit, wenn der 8250 keine Daten sendet oder empfängt, umgeschrieben werden.

4.6.9 Der Oszillator des 8250

Der interne Oszillatorteil des 8250 ist für einen Kristall entwickelt, der im fundamentalen Modus und paralleler Resonanz arbeitet. Tabelle 4-36 zeigt die erforderlichen Kristallparameter während Bild 4-53 die externe Beschaltung des Schwingkreises zeigt.

Wird eine externe Signalquelle verwendet, bleibt der Anschluß XTAL2 offen und das Signal wird dem Eingang XTAL1 zugeführt. Die Leistungsaufnahme reduziert sich bei der Verwendung einer externen Signalquelle um 50 Prozent. Die maximale Frequenz des 8250 ist 10 MHz, unabhängig von der Verwendung eines Kristalls oder einer externen Taktquelle. Somit beträgt die höchste Frequenz am Ausgang $\overline{\text{BAUDOUT}}$ 10 MHz und die maximale Baudrate 625 kHz.

Parameter	Wert
Frequenz	1,0 bis 10 MHz
Art	Parallel Resonanz, Fundamentalmodus
Kondensatoren	20 bis 32 pF
Widerstand	100 Ω (f=10 MHz; C=32 pF) 200 Ω (f=10 MHz; C=20 pF)

Tabelle 4-36. Technische Daten für die Beschaltung des Oszillators

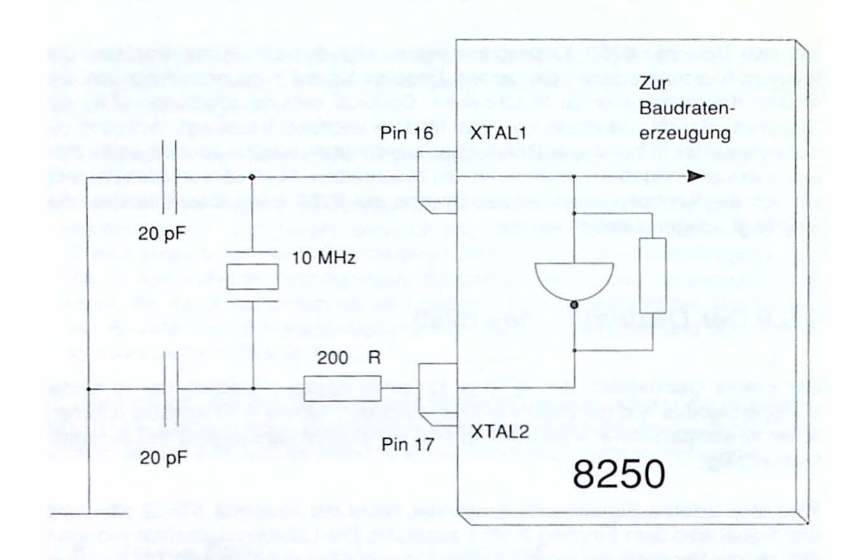


Bild 4-53. Die externe Beschaltung des Oszillators

4.6.10 Eine RS-232-C-Schnittstelle mit den Treibern 75150 und 75154

Eine recht häufig verwendete Schnittstelle zur seriellen Datenübertragung ist die RS-232-C-Schnittstelle nach dem EIA-Standard. Sie übersetzt einen digitalen High-Pegel des Computersystems mit einer Spannung von >2,8 V in eine Spannung von -3 V bis -15 V, und einen digitalen Low-Pegel von <0,8 V in eine Spannung von +3 V bis +15 V.

4.6.10.1 Der zweifache Leitungstreiber 75150

Es gibt eine Reihe integrierter Schaltungen, die in einem Chip diese Pegelwandlung vornehmen. Einer davon ist der Baustein 75150, der mit seinen Eingängen den digitalen Pegel von 0 V bis +5 V aufnimmt und ihn in den RS-232-C-Pegel übersetzt. Die Pin-Belegung findet sich in Bild 4-54. Er besitzt zwei Eingänge A1 und A2, an die die Ausgangssignale des Computersystems anzuschließen sind. Mit Hilfe des Freigabeeingangs G kann die Signalübertragung ermöglicht oder unterbunden werden. Der Eingang G ist high-aktiv, d.h. +5 V gibt die Eingänge A1 und A2 frei, Masse sperrt die Pegelumwandlung. In diesem Fall liegt an den Ausgängen Y1 und Y2 eine Spannung von +8 V an.

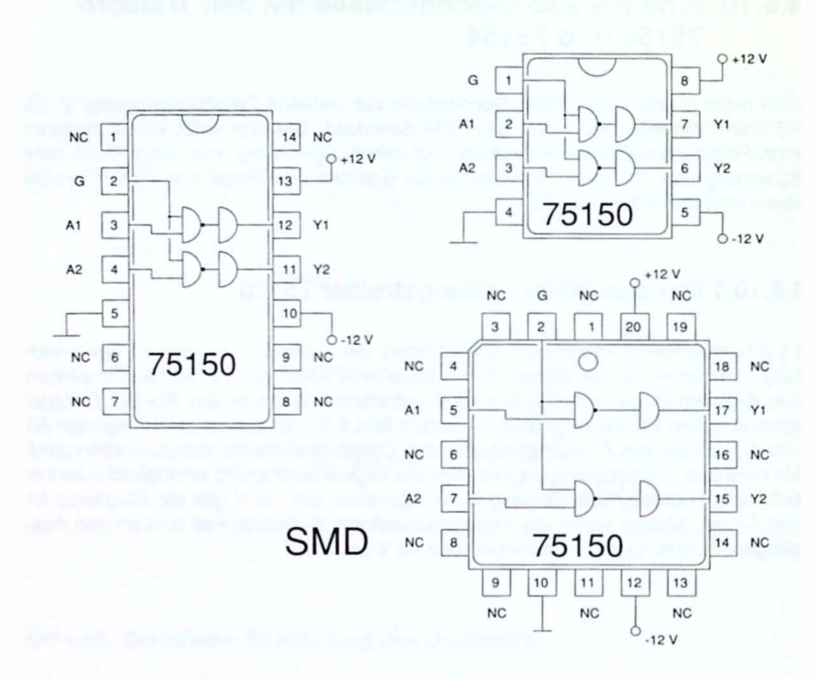


Bild 4-54. Pin-Belegung und Blockdiagramm des 75150

Der Baustein 75150 besitzt drei Stromversorgungsanschlüsse. Die Masseleitung des Computersystems verbinde man mit Pin 5, dem Masseanschluß. An Pin 13 lege man die positive, an Pin 10 die negative Betriebsspannung. Positive und negative Betriebsspannung sollten bezüglich Masse symmetrisch sein. Obwohl der Baustein für eine Maximalspannung von ±15 V an diesen Eingängen ausgelegt ist, empfiehlt sich für eine optimale Arbeitsweise die Spannung im Bereich von ±10,8 V bis ±13,2 V zu wählen, d.h. für die positive Spannung benutze man im Netzteil den Spannungsregler 7812, und für die negative Spannung den Spannungsregler 7912. Somit hat man als Betriebsspannung einen Wert von ±12 V. Unter diesen Voraussetzungen bewirkt ein Low-TTL-Pegel an den Eingängen A und bei freigegebenem Baustein (G an 0 V) an den Ausgängen Y eine Spannung von +8 V, bei High-TTL-Pegel an den Eingängen A eine Spannung von -8 V an den Ausgängen Y.

Die Strombelastung der TTL-Pegel durch die Eingänge des 72150 ist gering, sie beträgt maximal 15 µA. Die Ausgänge Y liefern typisch einen Strom von ±15 mA; der Kurzschlußstrom kann dabei auf ±30 mA ansteigen. Es sollte jedoch darauf geachtet werden, daß nicht beide Ausgänge zu derselben Zeit den Kurzschlußstrom liefern müssen.

Die Geschwindigkeit, mit der der Baustein den Pegel von +8 V nach 8 V und umgekehrt wechselt, ist entscheidend für die maximale Baudrate. Diese Schalteigenschaft hängt einerseits von der internen Struktur des 75150 ab und andererseits von der Länge der Datenverbindungsleitung und damit von der Kapazität und dem Widerstand, die eine lange Leitung mit sich führen kann. Typische Werte sind bei
U = ±12 V; C = 2500 pF; R = 3 - 7 kΩ eine Zeit von 1,4 µs und bei
U = ±12 V; C = 15 pF; R = 7 kΩ eine Zeit von 40 ns. Die Durchlaufverzögerung durch den 75150 beträgt 60 ns. Ein Vergleich der elektrischen Eigenschaften mit den geforderten Werten der RS-232-C-Schnittstelle ist in Tabelle 4-37 gegeben.

4.6.10.2 Der vierfache Leitungsempfänger 75154

Der 75154 ist das Gegenstück zum Leitungstreiber 75150. Er besitzt vier Empfangskreise A1 bis A4 mit einem Eingangswiderstand von 4,2 kΩ, die den RS-232-C-Pegel in einen TTL-Pegel umwandeln und ihn an den Ausgängen Y1 bis Y4 zur Verfügung stellen. Aus einer Spannung, die an den Eingängen zwischen -3 V und -15 V (kurzfristig auch -25 V) liegt, erzeugt der 75154 unabhängig von der benutzten Betriebsspannung einen TTL-High-Pegel an den Ausgängen von 3,5 V. Aus einer Spannung, die an den Eingängen zwischen +3 V und +15 V (kurzfristig auch +25 V) liegt, erzeugt der 75154 unabhängig von der benutzten Betriebsspannung einen TTL-Low-Pegel an den Ausgängen von 0,23 V. Eine Kontrolle über die internen Schmitt-Trigger hat man mit den Eingängen T1 bis T4. Liegen diese Eingänge an der positiven Versorgungsspannung, kippt der Baustein den Ausgangspegel bei einer Eingangsspannung im Bereich von 0 V und einer Hysteresis von ca. 3,3 V. Liegen diese Eingänge an Masse, befindet sich der Umschaltpunkt bei einer Eingangsspannung von ca. 1,4 V bis 2 V mit einer Hysteresis von ca. 0,8 V. Mit einem 10 k-Potentiometer zwischen der positiven Betriebsspannung und dem Eingang T kann in kritischen Fällen die Umschaltschwelle den besonderen Bedürfnissen angepaßt werden.

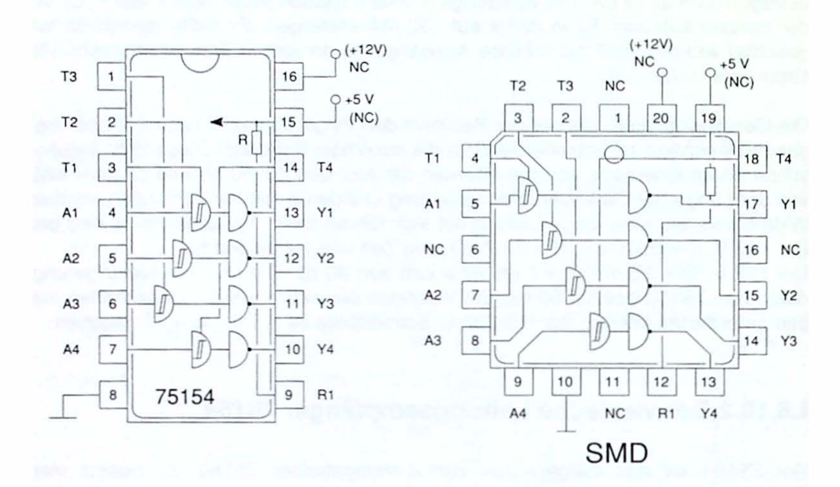


Bild 4-55. Pin-Belegung und Blockdiagramm des 75154

Im Gegensatz zum 75150 kommt dieser Baustein mit einer einzigen Versorgungsspannung aus, wobei man zwischen +5 V (Pin 15) und +12 V (Pin 16) wählen kann. Es ist nicht zulässig, beide Spannungen gleichzeitig anzuschließen.

Die Ausgänge können kurzfristig einen Kurzschlußstrom von 20 mA liefern und besitzen ein Fan-Out von 10. Die Durchlaufverzögerung des Signals beträgt 22 ns, die Zeit für einen Pegelwechsel an den Ausgängen liegt bei 9 ns.

4.6.10.3 Die RS-232-C-Schnittstelle (V.24)

Parameter	RS-232-C	Eigenschaften des 75150 bzw. 75154
Sendeform	Unbalanciert	Ausgang Y
Maximale Sendedistanz	15 m	Erweiterung ist mit einem Kondensator gegen Masse möglich
Maximale Sendegeschwindigkeit	20 kBaud	Mindestens 667 kBaud
		75150:
Maximale Ausgangsspannung	±25 V	±13,2 V
Minimale Ausgangsspannung	±5 V - ±15 V	±8 V
Maximaler Kurzschlußstrom	±500 mA	±30mA
Flankensteilheit	max. 30 V/µs	40 ns bzw. 20 ns
		75154:
Eingangswiderstand	3 kΩ- 7 kΩ	4,2 kΩ
Maximale Umschaltschwelle	±3 V	±2 V
Maximale Eingangsspannung	±25 V	±25 V

Tabelle 4-37. Die Bausteine 75150 und 75154 im Vergleich mit den Standardwerten der RS-232-C-Schnittstelle

Ein Beispiel für die serielle Kommunikation über die RS-232-C-Schnittstelle mit Hilfe zweier Kommunikationsbausteinen 8250 ist in Bild 4-56 gezeigt. Für einen Duplexbetrieb werden mindestens fünf Leitungen benötigt: +12 V, -12 V, TxD, RxD und Masse. Sie werden mit dem genormten 25poligem D-Subminiaturstecker nach Bild 4-57 verbunden. Die Steckverbindung in der oberen Hälfte besteht aus Stiften, die auf der Rückseite von einem Personal Computer als serielle Schnittstelle vorhanden ist. Die Signalbezeichnungen beziehen sich dabei auf die I/O-Einheit.

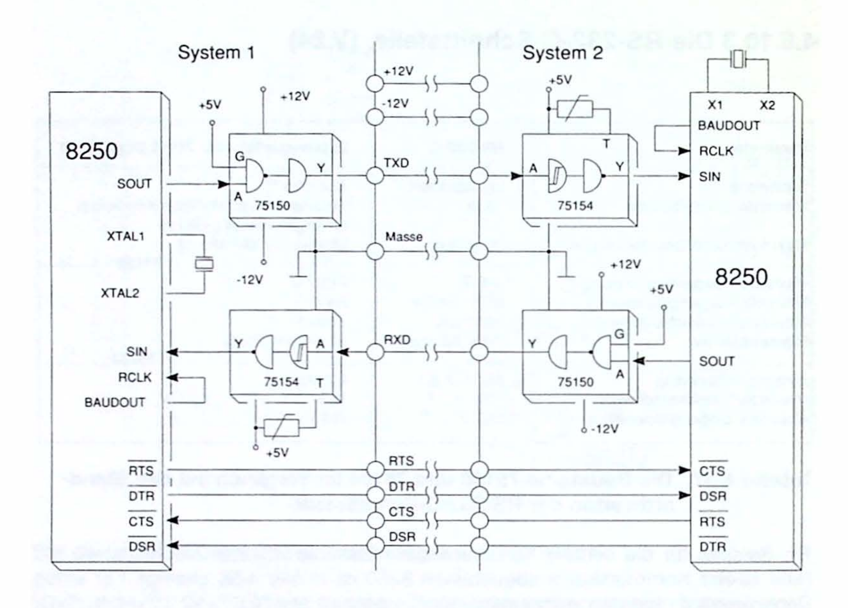


Bild 4-56. RS-232-C-Schnittstelle zwischen zwei Computersystemen mit dem seriellen Kommunikationsbaustein 8250

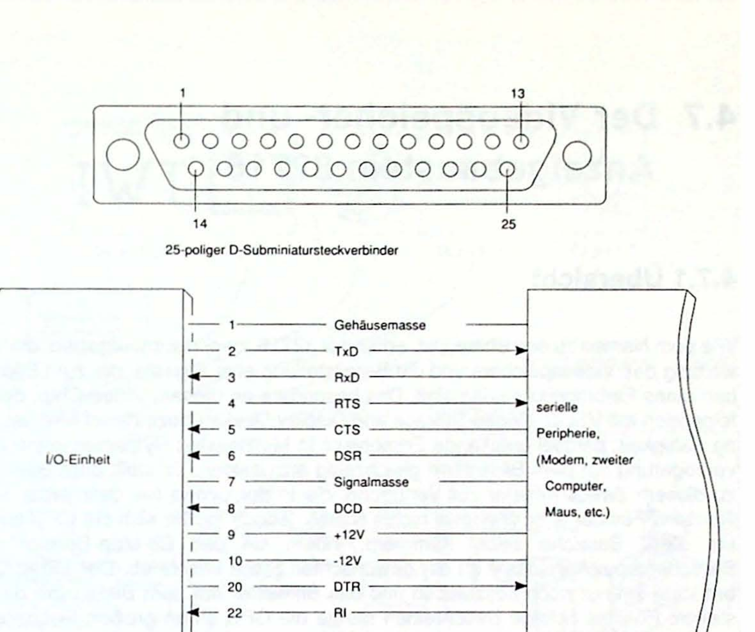


Bild 4-57. Anschlußbelegung der RS-232-C-Schnittstelle (V.24)

4.7 Der Videospeicher- und Anzeigebaustein 82716

4.7.1 Übersicht

Wie dem Namen zu entnehmen ist, erfüllt der 82716 zwei Hauptaufgaben: die Verwaltung des Videospeichers und die Bereitstellung aller Signale, die zum Bildaufbau eines Farbmonitors nötig sind. Das besondere an diesem Video-Chip, der im folgenden mit VSDD (Video Storage and Display Device) bezeichnet wird, ist seine Fähigkeit, parallel anfallende Ergebnisse in Multitasking-Systemen ohne Zeitverzögerung auf dem Bildschirm gleichzeitig anzuzeigen. Er stellt dem Benutzer zu diesem Zweck Fenster zur Verfügung, die in der Größe frei definierbar sind. Nun sind Fenster oder Windows nichts Neues; jedoch mußte sich die CPU bisher um diese Bereiche selbst kümmern, indem sie den Bit-Map-Bereich des Bildschirmspeichers direkt an der gewünschten Stelle beschrieb. Der Video-Chip brauchte ihn nur noch auszulesen und das Bitmuster auf dem Bildschirm darzustellen. Für das richtige Beschreiben mußte die CPU einen großen Rechenaufwand betreiben, der nicht gerade zur Beschleunigung der Prozessorgeschwindigkeit beitrug.

Anders sind die Verhältnisse in Systemen mit dem VSDD als Video-Chip. Der VSDD kümmert sich ohne Zutun der CPU um die Größe und Lage, auf Wunsch auch um die Bewegung der Fenster, die wohl richtiger mit Objekte zu bezeichnen sind. Er steuert bis zu 16 dieser Objekte gleichzeitig auf dem Bildschirm, wobei die Größe eines Objekts durchaus die Größe des Bildschirms überschreiten und nach unten hin beliebig klein sein kann. Die Objekte können als Bit-Map oder Charakter-Felder definiert werden, womit eine Einteilung der Fenster vorgenommen wird. Der Inhalt und die Position eines jeden Objekts kann unabhängig von den anderen geändert werden.

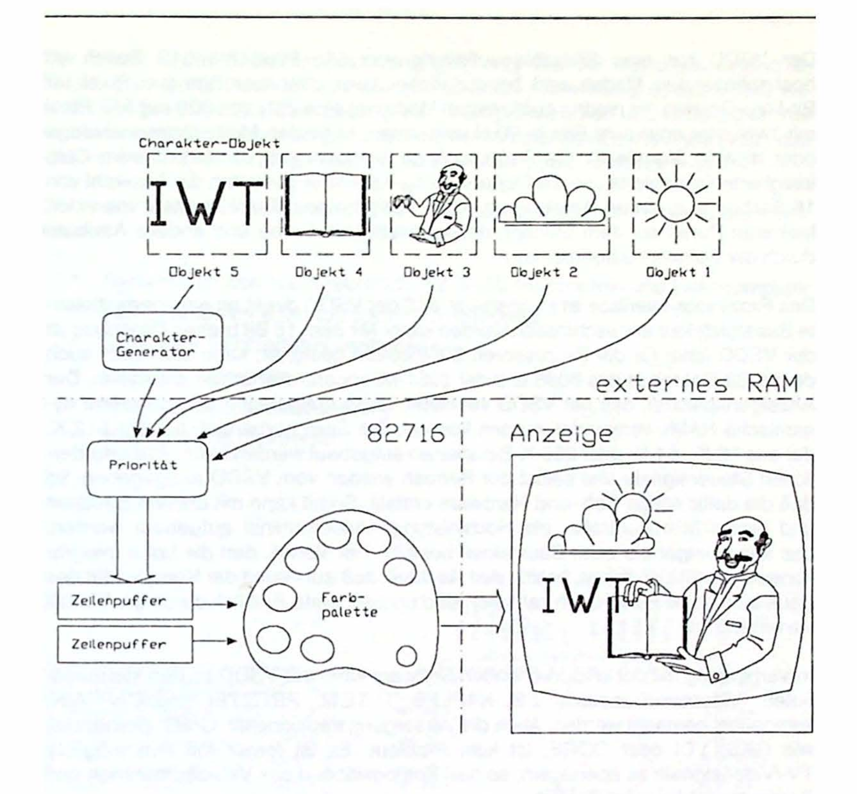


Bild 4-58. Der Bildaufbau des VSDD 82716

Es gibt für ihn also keinen Unterschied wischen Text- und Grafikmodus; alles was er darstellt, befindet sich im Grafikmodus, also auch der Text, für dessen Wiedergabe der VSDD auf zwei Charakter-Generatoren mit je 256 Zeichen zugreifen kann. Diese Charakter sind vom Anwender definierbar und können durch die System-Software dynamisch geändert werden. Es eröffnen sich hervorragende Möglichkeiten zur Textverarbeitung. Die Darstellung von Proportionalschrift, doppelt hoher Zeichen oder Ornamentschrift auf dem Bildschirm ist damit kein Problem mehr. Man braucht dazu nur die beiden Charakter-Generatoren mit den gewünschten Daten zu laden.

Der VSDD hat eine Bildschirmauflösung von 640 Pixel mit 512 Zeilen im hochauflösenden Modus und benutzt dabei zwei oder vier Bits pro Pixel für Bit-Map-Objekte. Im niedrig auflösenden Modus ist eine Zahl von 320 auf 512 Pixel mit zwei, vier oder acht Bits je Pixel vorhanden. In beiden Modi können analoge oder digitale Signale für die Farbausgänge programmiert werden. Die im Chip integrierte Farbpalette und die Digital-Analog-Konverter gestatten die Auswahl von 16 Farben aus einem Bereich von 4096. (Unter einem Pixel versteht man den kleinsten Punkt auf dem Monitor, dessen Helligkeit, Farbe und andere Attribute durch die Software steuerbar ist.)

Das Prozessor-Interface ist so gestaltet, daß der VSDD direkt an eine gemultiplexte Busarchitektur angeschlossen werden kann. Mit dem 16 Bit breiten Datenbus ist der VSDD ideal für die Prozessoren 8086/80286 geeignet, kann sich aber auch dem 8-Bit-Datenbus des 8088 und der 8051-Mikrocontrollerfamilie anpassen. Der Anzeigenspeicher, den der VSDD verwaltet, ist so ausgelegt, daß preiswerte dynamische RAMs verwendet werden können. Die Speicherbereich beträgt 512 K, der aus 16-K-, 64-K- oder 256-K-Bausteinen aufgebaut werden kann. Alle erforderlichen Steuersignale und selbst der Refresh werden vom VSDD ausgegeben, so daß die dafür nötige Soft- und Hardware entfällt. Somit kann mit diesem Baustein und einem Mikrocontroller ein Hochleistungs-Videoterminal aufgebaut werden, das aus weniger als zehn Bausteinen besteht. Der Vorteil, den die hohe Integrationsdichte mit sich bringt, besitzt den Nachteil, daß auf Grund der Komplexität des Bausteins seine Funktionen zahlreich sind und der erste Einblick durch die Vielfalt verwirrend ist.

In Verbindung mit der entsprechenden Software kann der VSDD zu den internationalen Videotex-Standards, z.B. NAPLPS, TETETL, PRESTEL und CAPTAIN, kompatibel gemacht werden. Auch die Versorgung traditioneller Grafik-Standards, wie GKS, VDI oder CORE, ist kein Problem. Es ist ferner mit ihm möglich, TV-/Videosignale zu überlagern, so daß Kompositionen aus Videoaufnahmen und Computer-Bilder möglich sind.

Die augenblicklichen europäischen Videotex-Standards basieren auf Systemen wie PRESTEL und TELETEL. Sie benutzen alpha-geometrische Standards mit entweder seriellen oder parallelen Attributen. Der VSDD gestattet den Gebrauch von zwei Arten an Charakter-Objekten, entweder ein oder drei Bytes je Charakter. Die Wahl für ein Byte je Charakter ist dann sinnvoll, wenn Standard-Computerausgänge vorhanden sind oder wenn der freie RAM-Platz ein Problem ist. Dieses Byte besteht dann meist aus dem ASCII-Kode des darzustellenden Zeichens. Drei Bytes je Charakter sind für Videotex-Anwendungen zu empfehlen, da damit für jedes Charakter-Zeichen spezielle Attribute wie Farbe, Blinken, sichtbar/unsichtbar, doppelte Höhe oder Breite etc. wählbar sind. Der Inhalt des Charakter-Generators ist jederzeit änderbar, z.B. abhängig vom Programm, da er Bestandteil des RAM-Bereichs ist.

Der Trend in der Bildschirmgestaltung geht weg von der alpha-geometrischen Phase zur rein grafischen Darstellung. Für diese Anwendung ist der VSDD bestens ausgestattet. Die Übertragung von Grafikprogrammen und Grafikdaten von nicht kompatiblen Systemen geschieht mit dem Virtual Device Interface (VDI), das die heute üblichen Grafikprogamme auch für den VSDD lauffähig macht.

Die Anwendungsmöglichkeiten des 82716 lassen auf folgende Punkte zusammenfassen:

- Systeme für den Hausgebrauch: TV, VCR, Videospiele und Heimcomputer
- Alphanumerische Monochrom/Farb-Terminals
- Geräte für Echtzeitüberwachungen
- Videotex-Terminals
- Anzeige im Verkehrsbereich
- Medizinische Elektronik

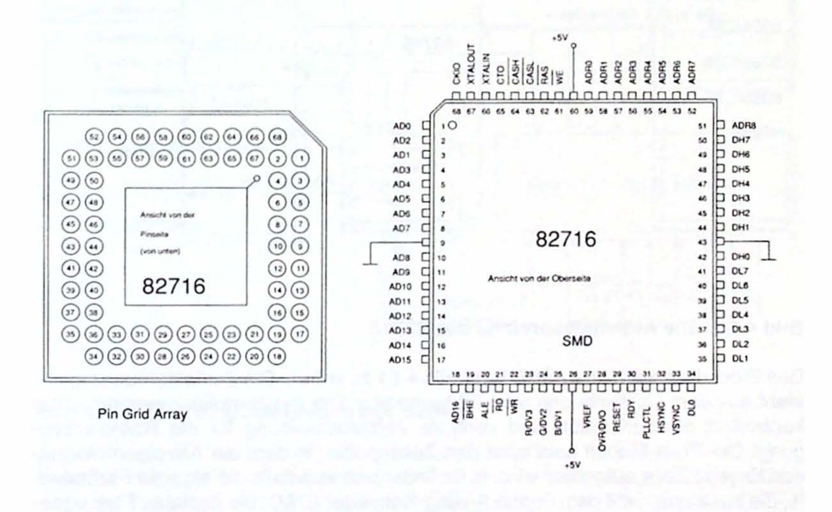


Bild 4-59. Die Pin-Belegung des 82716

4.7.2 Die Architektur des 82716

Bild 4-60 zeigt, daß die Aktivitäten des VSDD in drei Bereiche gegliedert sind:

1. Die Zusammenarbeit mit der CPU, von der er Befehle und Daten entgegen nimmt.
2. Der Zugriff auf den VSDD-eigenen Speicher, in dem sich die Steuerregister, die Informationen über die Objekte, die beiden Charakter-Generatoren, die Farbtabelle und die Daten für die Objekte befinden.
3. Die Ansteuerung des Monitors mit RGB- und Synchronisationssignalen.

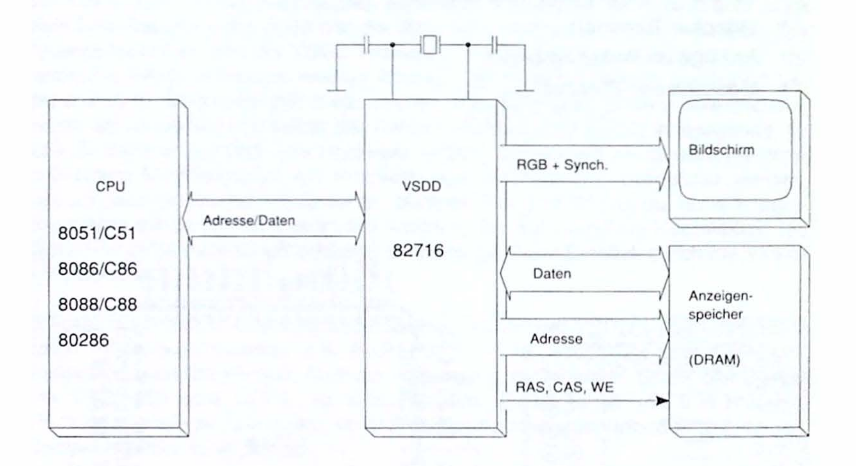


Bild 4-60. Die Aktivitätsbereiche des 82716

Das Blockdiagramm des VSDD ist in Bild 4-61 zu sehen. Die Zeittaktsteuerung besteht aus dem Oszillator und dem Taktgenerator. Der Synchronisierungsgenerator kontrolliert die horizontale und vertikale Zeittaktsteuerung für die Rastererzeugung. Die Pixel-Einheit beinhaltet den Zeilenpuffer, in dem die Anzeigeninformation für jede Zeile aufgebaut wird. In ihr findet sich ebenfalls die aktuelle Farbtabelle, die zusammen mit dem Digital-Analog-Konverter (DAC) die digitalen Farbsignale in ein analoges RGB-Signal für den TV-Monitor umwandelt. Die Speicher- und die Bus-Interface-Einheit stellen jeweils die Verbindung zum Speicher bzw. zur CPU her.

Der 82716 teilt seinen Speicher in zwei Segmente ein (Bild 4-62), von denen das Registersegment im Vergleich zum Datensegment einen vergleichsweise winzigen Teil ausmacht. Die 32 Bytes im Registersegment stellen dem VSDD Informationen über die Systemkonfiguration, die Basisadresse der Datenblocks und über das Adreßfenster, über das die CPU auf den VSDD-RAM-Bereich zugreifen kann, zur Verfügung.

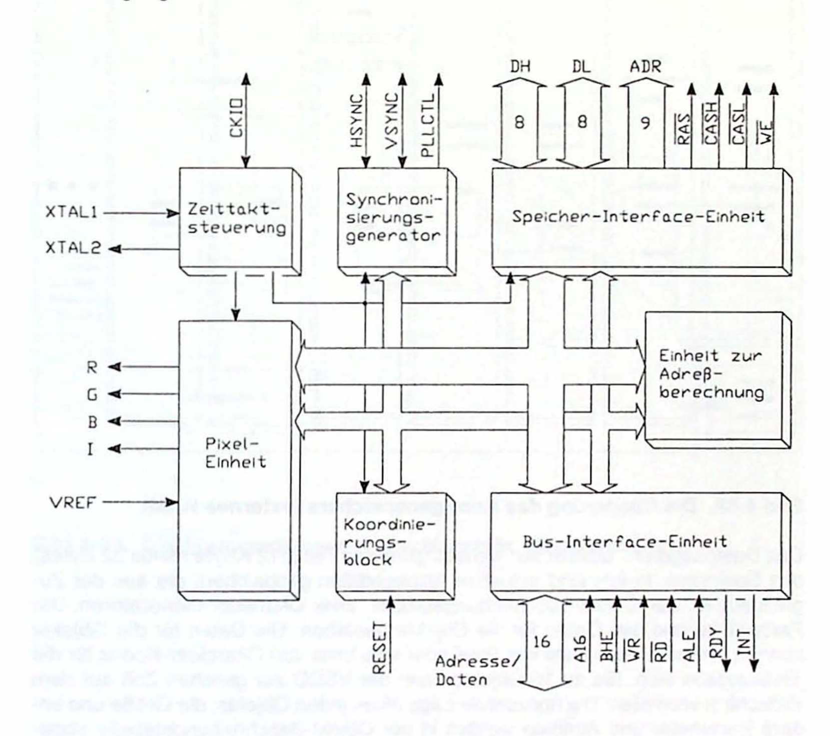


Bild 4-61. Das Blockdiagramm des 82716

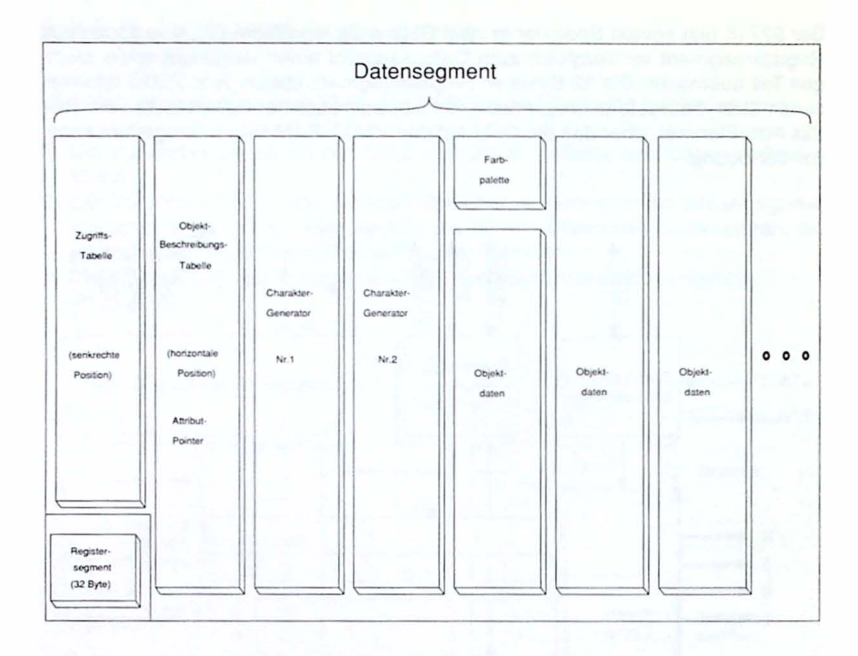


Bild 4-62. Die Gliederung des Anzeigenspeichers (externes RAM)

Das Datensegment umfaßt den weitaus größeren Teil (512 KByte minus 32 Bytes) des Speichers. In ihm sind aktuellen Anzeigedaten gespeichert, die aus der Zugriffstabelle, der Objekt-Beschreibungstabelle, zwei Charakter-Generatoren, der Farbpalette und den Daten für die Objekte bestehen. Die Daten für die Objekte können entweder eine Liste von Pixel oder eine Liste von Charakter-Kodes für die Textausgabe sein. Bis zu 16 Objekte kann der VSDD zur gleichen Zeit auf dem Bildschirm verwalten. Die horizontale Lage eines jeden Objekts, die Größe und andere Parameter und Attribute werden in der Objekt-Beschreibungstabelle abgelegt, wohingegen die senkrechte Position sich in der Zugriffstabelle befindet.

Die RGB-Signale werden durch drei interne Digital-zu-Analog-Konvertern (DAC) erzeugt, deren 4-Bit-Eingänge über die Farbtabelle versorgt werden. In dieser Tabelle sind 16 programmierbare Mixturen der Farben Rot, Grün und Blau (RGB). Eine Auswahl der Farbe wird für jedes Pixel mit einer 4-Bit-Spezifikation in den Zeilenpuffern für jede Zeile der Anzeige getroffen.

Der VSDD besitzt zwei komplette Zeilenspeicher. Während der eine gerade zur Anzeige kommt, wird der andere mit den errechneten Werten (Farbe, Objekt, Priorität etc.) gefüllt (Bild 4-63). Die Steuerung erfolgt durch den Koordinierungsblock. Im Koordinierungsblock befindet sich ein kleines Maschinenprogramm, dessen Algorithmus Bild 4-64 wiedergibt.

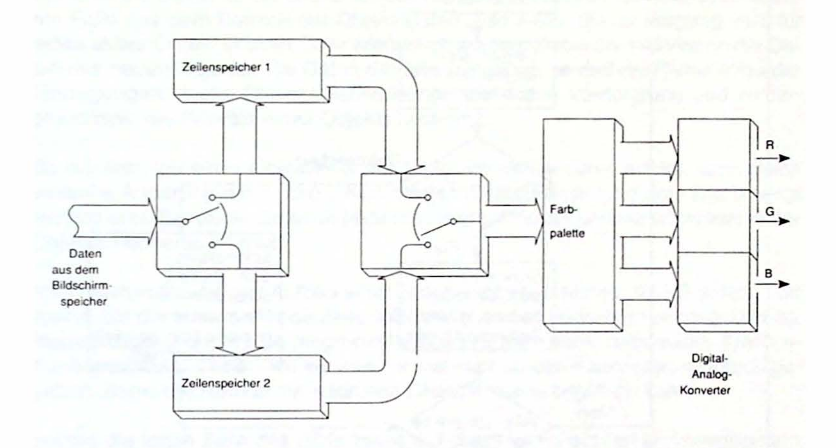


Bild 4-63. Die Erzeugung einer Bildschirmzeile durch den 82716

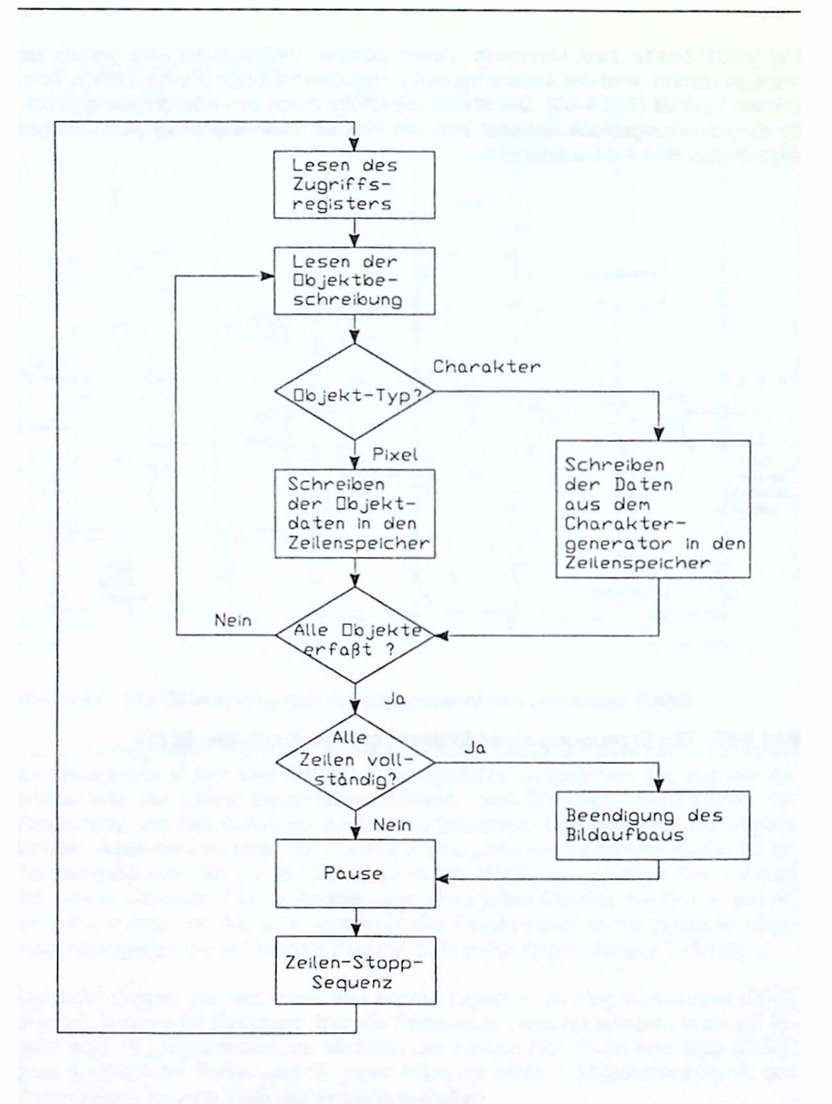


Bild 4-64. Flußdiagramm für die Konstruktion einer Bildschirmzeile

Zu Beginn einer Bildschirmzeile ließt der VSDD die Eintragung in der Zugriffstabelle für die zu konstruierende Zeile und fragt nach jedem Objekt, das in dieser Zeile dargestellt werden soll. Für anwesende Objekte wird der entsprechende Datenblock aus der Objekt-Beschreibungstabelle gelesen, der alle Informationen wie Objekt-Typ, Breite, horizontale Position etc. enthält. Es gibt zwei Objekt-Typen: Pixel- (Grafik) und Charakter-Objekte (Text). Im Falle eines Charakter-Objekts kommen die Daten für die Bildschirmzeile aus dem Charakter-Generator, im anderen Falle aus dem Bereich der Objekt-Daten (Bild 4-62). Dieser Vorgang muß für jedes aktive Objekt in einer Zeile wiederholt werden. Dabei überschreiben die Daten des neuen Objektes die Daten des alten Objektes, so daß die Reihenfolge der Eintragungen in der Objekt-Beschreibungstabelle den Vordergrund und Hintergrund bzw. die Priorität eines Objekts bestimmt.

Da die Anzeige eines Objektes unabhängig von den anderen erfolgt, kann durch einfache Änderung der x- und y-Koordinaten dieses Fenster auf dem Bild bewegt werden und überdeckt dabei Objekte niedrigerer Priorität und verschwindet hinter Objekten höherer Priorität.

Nach dem vollständigen Aufbau einer Bildschirmzeile bleibt der VSDD untätig und wartet, bis die vorangehende Zeile vollständig an den Bildschirm ausgegeben ist. Diese Ausgabe endet eine programmierbare Zeit nach dem horizontalen Synchronisationsimpuls. Dabei wird ein interner Interrupt an den Koordinierungsblock gegeben, damit der Aufbau der nächsten Bildschirmzeile beginnen kann.

Sobald die letzte Zeile des Bildschirms auf diese Art konstruiert und ausgegeben wurde, wird eine Bild-Stopp-Sequenz ausgeführt, nach der der VSDD untätig auf ein Signal vom Koordinierungsblock wartet, um danach mit der ersten Zeile des folgenden Bildschirmaufbaus zu beginnen.

Die CPU hat jederzeit Zugriffsmöglichkeit auf den Bildschirmspeicher. Allerdings kann der VSDD so programmiert werden, daß er der CPU während einer Zeilenkonstruktion nur eine beschränkte Zahl an Zugriffen erlaubt, da er ja selbst Daten aus dem Bildschirmspeicher holen muß. Sollte dennoch wegen der Vielzahl der Objekte der Aufbau einer Bildschirmzeile durch die Eingriffe der CPU in Frage gestellt sein, meldet das der VSDD bei der CPU durch einen Interrupt an, so daß sie Ihre Aktivitäten in Richtung VSDD bremsen kann.

4.7.3 Die Hardware des 82716

Der Teil im VSDD, der mit der CPU zusammenarbeitet, wird Bus-Interface-Einheit genannt. Über diese Einheit hat der Prozessor die Möglichkeit, auf den Anzeigenspeicher zuzugreifen. Er kann ihn beschreiben und lesen, oder ihn als allgemeinen Datenspeicher benutzen, wenn der 82716 diesen Teil des RAMs nicht benötigt. Die Bus-Interface-Einheit besteht aus einem gemultiplexten 16-Bit-Adreß-/Datenbus und einen programmierbaren Chip-Select-Eingang (A16) und verfügt über die Kontrollsignaleingänge ALE, $\overline{RD}$, $\overline{WR}$ und BHE. Für die Rückkopplung ist der Bereitschaftsausgang RDY vorhanden. Die Speicher-Interface-Einheit koordiniert den Zugriff und den Refresh des externen DRAMs. Sie besteht aus einem gemultiplexten 9-Bit-Reihen- und -Spalten-Adreßbus, einem separaten 16-Bit-Datenbus mit den entsprechenden Kontrollsignalen $\overline{RAS}$, $\overline{CAS}$ und $\overline{WE}$.

Der programmierbare Synchronisierungsgenerator gibt die vertikalen und horizontalen Synchronisationsimpulse für den Monitor aus. Die Zahl der Zeilen für das Bildschirmraster, die Art des Rasters (verzahnt oder nicht verzahnt), die Periode und Breite des Synchronisierungsimpulses etc. sind alle programmierbar. Der Synchronisierungsgenerator kann darüber hinaus extern erzeugte Synchronisierungssignale verarbeiten. Der Ausgang PLLCTL ist vorgesehen, um den Oszillator für den Videotakt mit einer Phase-Locked-Loop-Operation (PLL) zu steuern.

4.7.3.1 Die Zeittaktsteuerung

Die Einheit zur Zeittaktsteuerung (Bild 4-65) verfügt über einen internen Oszillator und die Takterzeugung für die interne Logik und den Videobereich. Auf Wunsch kann der Pin CKIO (Clock Input/Output) das intern erzeugte Taktsignal ausgeben oder ein externes empfangen. Die externe Beschaltung des Schwingkreises erfolgt an den Pins XTAL1 und XTAL2. Bild 4-66 zeigt einen 14,5-MHz-Quarz mit zwei Kondensatoren der Kapazität 30 pF ±10 pF. Die Wahl dieser Werte ermöglicht eine maximale Geschwindigkeit bei der Zeilenkonstruktion und beim RAM-Zugriff.

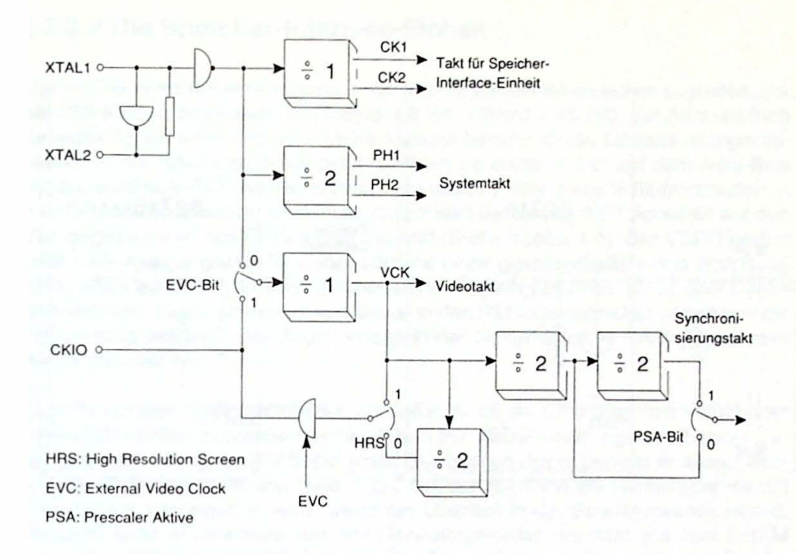


Bild 4-65. Der Aufbau der internen Zeittaktsteuerung

Darüber hinaus kann ein externer Oszillator, angeschlossen an XTAL1, benutzt werden. XTAL2 bleibt in diesem Fall offen. Die einzige Forderung an das externe Signal ist ein Tastverhältnis von 50 Prozent.

Sollen mit dem VSDD Videobilder überlagert werden, empfiehlt sich die Verwendung eines LC-Oszillators an den Pins XTAL1 und XTAL2, da mit diesem Schwingkreis und dem Ausgang PLLCTL leichter eine Frequenzvariation möglich ist, wenn der VSDD mit einem externen Signal synchronisiert werden soll.

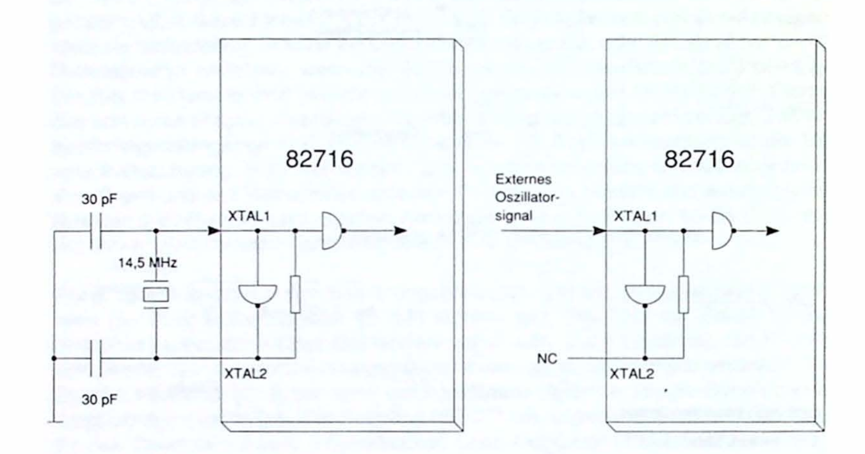


Bild 4-66. Die Oszillator-Konfiguration des VSDD

In jedem Fall wird das Signal am Eingang XTAL1 dem Taktgenerator der Speicher-Interface-Einheit zugeführt. Dasselbe Signal treibt ferner den Videotaktgenerator und nach entsprechender Programmierung auch den Synchronisierungsgenerator (EVC-Bit = 0). In diesem Fall liegt am Ausgang CKIO ein Signal an, das bei hochauflösendem Bildschirm (High Screen Resolution HSR-Bit = 1) dieselbe Frequenz oder die Hälfte der Frequenz (HSR-Bit = 0) aufweist. Das CKIO-Signal wird in der Regel dann benutzt, wenn der VSDD unter Umgehung der Digital-Analog-Konverter digitale Videoinformationen ausgibt. In diesem Fall zeigt das CKIO-Signal das Vorliegen gültiger digitaler Informationen an.

Ist das External-Video-Clock-Bit (EVC) gesetzt, werden der Videotakt- und der Synchronisierungsgenerator durch ein externes Signal gesteuert, das am Pin CKIO anliegt. Der Pin CKIO ist in diesem Fall als Eingang konfiguriert.

4.7.3.2 Die Speicher-Interface-Einheit

Der VSDD kann auf einen Bereich von 512 KByte DRAM-Speicher zugreifen, der als 265 KWord organisiert ist (1 Byte = 8 Bit; 1 Word = 16 Bit). Zur Adressierung für jedes 16-Bit-Wort wird eine 18-Bit-Adresse benutzt. Diese Adresse wird gemultiplext an die RAM-Bausteine geführt. Wenn die ersten 9-Bits auf dem Adreßbus liegen, wird das $\overline{RAS}$-Signal aktiviert, mit dessen Hilfe diese 9-Bit-Information in den DRAMs gespeichert wird. Kurz darauf wird der zweite 9-Bit-Adreßteil auf den Bus gegeben und das $\overline{CAS}$-Signal aktiviert (Siehe Kapitel 1.4). Der VSDD besitzt zwei $\overline{CAS}$-Ausgänge, $\overline{CASL}$ und $\overline{CASH}$, die beide gleichzeitig für einen Worttransfer aktiviert werden, während bei einem Byte-Transfer nur eines, $\overline{CASL}$ oder $\overline{CASH}$ aktiviert wird. Beim Schreiben von Daten in den Bildschirmspeicher wird ferner die $\overline{WE}$-Leitung aktiviert. Der Zugriff erfolgt in der für dynamische RAMs gewohnten Weise (Kapitel 1.4).

Zugriffe auf den Speicher können entweder durch die CPU über den VSDD oder vom VSDD selbst ausgelöst werden. Wenn der VSDD für die eigenen Zwecke Daten aus dem DRAM ließt, z.b. bei einer Zeilenkonstruktion, benutzt er ausschließlich die Seitenadressierung (Bild 1-25), bei der nur dann die Reihenadresse mit $\overline{RAS}$-Signal ausgegeben wird, wenn ein Überlauf in der Spaltenadresse eintritt. Dadurch kann er innerhalb von drei Oszillatorperioden ein Wort aus dem DRAM lesen. Der 82716 verfügt über ein internes Bit, mit dessen Hilfe er auf schnelle oder langsame dynamische RAMs einstellbar ist. Es handelt sich um das Slow-Access-Bit (SAB). Ist es gesetzt, werden drei, ist es gelöscht, werden zwei Oszillatorperioden für einen Lesevorgang benötigt. Auch der Schreibvorgang ist damit um eine Periode verkürzbar.

Beispiel: Bei einer Taktfrequenz von 14,5 MHz beträgt die Taktperiode 70 ns, so daß eine Zugriffszeit von 140 ns oder 210 ns wählbar ist, und es können DRAM-Bausteine mit einer Zugriffszeit von 140 ns oder weniger benutzt werden.

4.7.3.2.1 Die DRAM-Konfiguration

Die als Hardware angeschlossene DRAM-Konfiguration muß dem 82716 in der Initiierungssequenz mitgeteilt werden. Dafür stehen dem VSDD vier Bits zur Verfügung: DS0, DS1, DOF und SAB.

Die Buchstaben DS stehen für DRAM Size und geben dem VSDD Informationen über die Größe des zur Verfügung stehenden RAMs (16 K, 64 K oder 256 K), denn es muß nicht in jedem Fall der maximale Speicher vorhanden sein. DOF steht für DRAM Organization Flag und hat Einfluß auf die Verwaltung des Speichers. Sein Inhalt zeigt dem VSDD an, ob Nibble- oder Bit-Speicher vorhanden sind. Nähere Informationen gibt Tabelle 4-38. Das Bit SAB unterscheidet zwischen schnellen und langsamen DRAM-Bausteinen. Diese Informationen sind für den 82716 sehr wichtig, da sie über den Gebrauch der Adreß- und Steuerleitungen entscheiden.

DS1	DS0	DOF	DRAM-Konfiguration	max. Größe	Aktive Reihe	Aktive Spalte	Adreß-Pins Bankauswahl
0	0	0	16 K x 1	32 KByte	0-6	0-6	keine
0	0	1	16 K x 4	128 KByte	0-7	0-5	mit 6,7 bei $\overline{CAS}$
0	1	X	64 K x 1 oder x 4	512 KByte	0-7	0-7	mit 8 bei $\overline{RAS}$, $\overline{CAS}$
1	X	X	256 K x 1	512 KByte	0-8	0-8	keine

Tabelle 4-38. Die Festlegung der RAM-Größe

Für 16 K x 1 DRAMs wird eine 7-Bit-Reihen und -Spaltenadresse auf den Leitungen ADR0 bis ADR6 erzeugt; die Leitungen ADR6, ADR7 und ADR8 werden nicht benutzt. Somit werden 16 solcher Bausteine benötigt. Die Verwendung dieser Speicher ist in Bild 4-67 gezeigt.

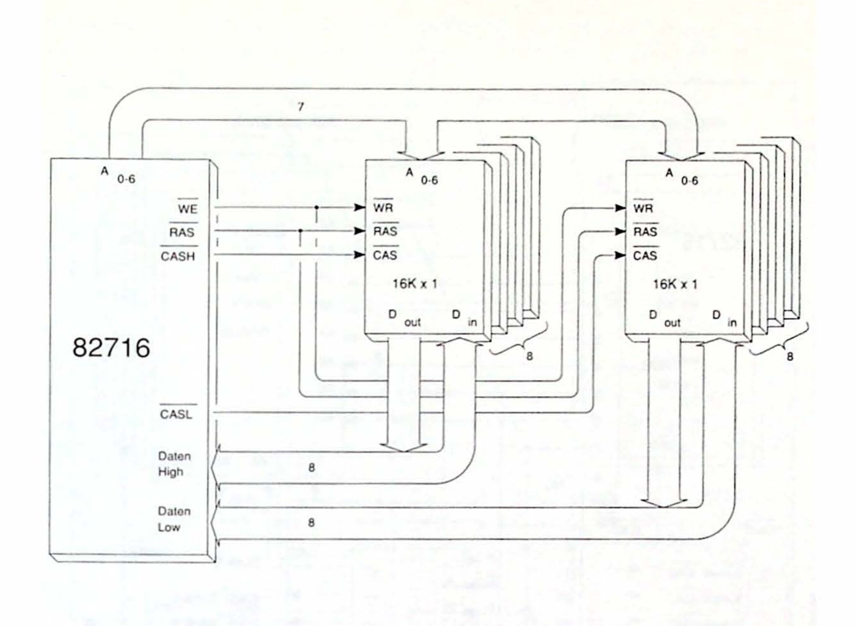


Bild 4-67. Die Verwendung von 16 K x 1 DRAM-Bausteine als Bildschirmspeicher

Für Bausteine der Organisation 16 K x 4 wird eine 8-Bit-Reihenadresse an den Ausgängen ADR 0-7 ausgegeben, gefolgt von einer 6-Bit-Spaltenadresse an den Leitungen ADR 0-5. Die Leitungen ADR 6 und ADR 7 geben zusammen mit der Spaltenadresse die Bank-Select-Bits (ADR 7 = MSB, ADR 6 = LSB) aus. Somit werden mindestens vier Bausteine dieser Art benötigt, um die erforderliche Bitbreite von 16 zu erreichen. Zusätzlich kann eine Dekodierung der Bank-Select-Bits erfolgen, so daß mit 4 Bänken ein Bereich von 64 K erreichbar ist (Bild 4-68).

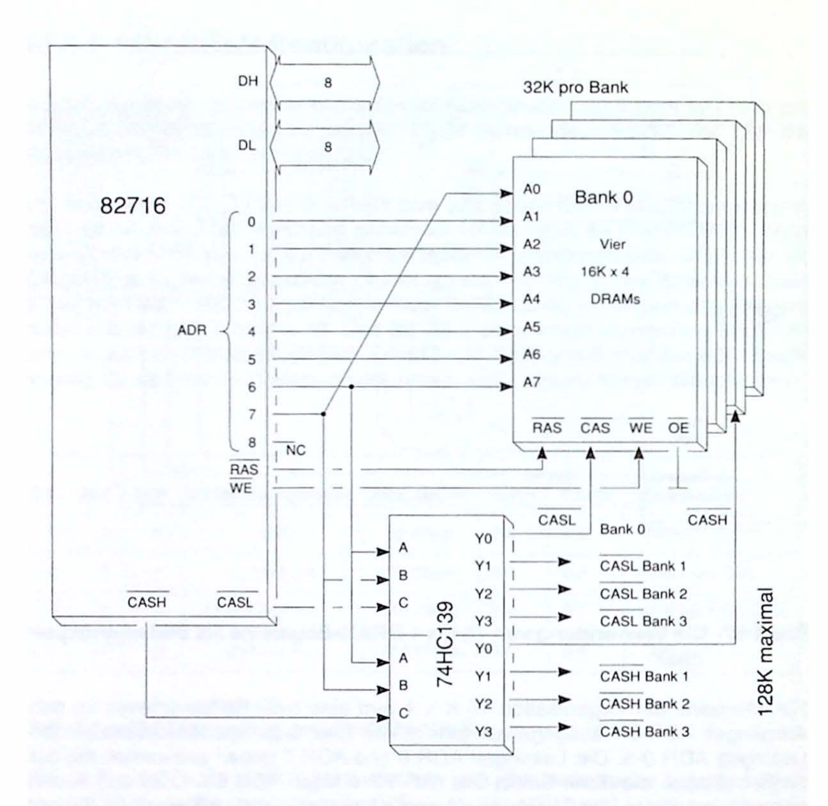


Bild 4-68. Bildschirmspeicher mit 16 K x 4 DRAM-Bausteine

Das Schaltbild in Bild 4-68 benutzt die $\overline{\text{CAS}}$-Eingänge des DRAMs als Chip-Select-Eingänge. Die $\overline{\text{OE}}$-Eingänge sind fest mit Masse verbunden. Des weiteren benötigen Standard 16 K x 4 DRAMs die Spaltenadressen an den Eingängen 1 - 6, während sie der VSDD an den Ausgängen 0 - 5 ausgibt. Aus diesem Grund werden die Ausgänge ADR 0 - 6 mit den Eingängen A 1 - 7 des DRAMs, und ADR 7 mit A0 verbunden.

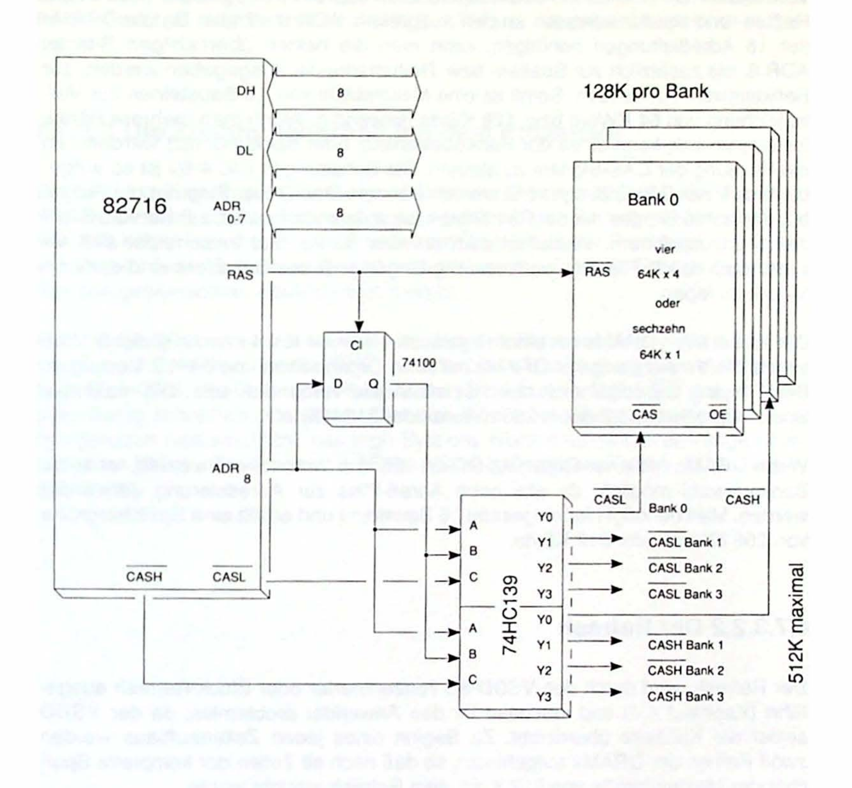


Bild 4-69. Bildschirmspeicher mit 64 K x 1 oder 64 K x 4 DRAM-Bausteine

Verwendet man DRAMs mit einer Organisation von 64 K x 1, gibt der 82716 acht Reihen- und Spaltenadressen an den Ausgängen ADR 0 - 7 aus. Da die DRAMs nur 16 Adreßleitungen benötigen, kann man die beiden überzähligen Bits an ADR 8, die zusätzlich zur Spalten- bzw. Reihenadresse ausgegeben werden, zur Bankauswahl verwenden. Somit ist eine Minimalzahl von 16 Bausteinen zur Verwirklichung von 64 KWord bzw. 128 KByte notwendig. Wenn man mehrere Bänke installieren will, kann eines der Bankauswahlbits oder beide benutzt werden, um die Richtung der $\overline{CAS}$-Signale zu steuern. Die Schaltung in Bild 4-69 ist so aufgebaut, daß vier Bänke ausgewählt werden können. Das D-Flip-Flop hat die Aufgabe, das achte Bit, das mit der Reihenadresse ausgegeben wird, zur Bankauswahl zwischenzuspeichern. Verzichtet man auf vier Bänke und entscheidet sich für zwei, kann das D-Flip-Flop entfallen; die Eingänge B des Dekoders sind dann an Masse zu legen.

Der Einsatz von DRAMs mit einer Organisation von 64 K x 4 kann in gleicher Weise wie die Verwendung von DRAMs mit einer Organisation von 64 K x 1 erfolgen. Der Eingang $\overline{OE}$ sollte auch hier fest mit Masse verbunden sein. Die maximale Speichergröße beträgt dabei 256 KWord oder 512 KByte.

Wenn DRAMs mit einer Organisation von 256 K x 1 verwendet werden, ist keine Bankauswahl möglich, da alle neun Adreß-Pins zur Adressierung verwendet werden. Man benötigt hier insgesamt 16 Bausteine und erhält eine Speichergröße von 256 KWord oder 512 KByte.

4.7.3.2.2 Der Refresh

Der Refresh wird durch den VSDD als konzentrierter oder Block-Refresh ausgeführt (Kapitel 1.4.4) und erscheint für den Anwender problemlos, da der VSDD selbst die Kontrolle übernimmt. Zu Beginn eines jeden Zeilenaufbaus werden zwölf Reihen der DRAMs aufgefrischt, so daß nach elf Zeilen der komplette Speicher der Maximalgröße von 512 K mit dem Refresh erreicht wurde.

Am Ende des kompletten Bildaufbaus legt der 82716 eine Ruhepause von 740 Oszillatorperioden ein, die er braucht, um die internen Register zu aktualisieren. Wird er in dieser Zeit durch Anforderungen von der CPU unterbrochen, kann die dafür erforderliche Zeit unter schlimmsten Bedingungen auf 8880 Oszillatorperioden steigen. Bei einem 14,5-MHz-Oszillator dauert die Periode 70 ns, so daß die Maximalzeit 621,6 µs betragen kann. Auch das ist weiter nicht Besorgnis erregend, da das erforderliche Refresh-Intervall abhängig vom Baustein 2000 µs bzw. 4000 µs lang sein kann.

Die meisten Bildschirme benötigen eine Zeit für den Zeilenaufbau von weniger als 64 µs. Somit ist mit maximal elf Zeilen (t = 704 µs) der gesamte Speicher aufgefrischt.

4.7.3.3 Die Zusammenarbeit mit dem Prozessor

Die CPU hat Zugang zum Bildschirmspeicher nur über den 82716. Dafür bietet der VSDD 17 Adreß- und 16 Datenleitungen an. Mit 17 Adreßleitungen ist ein Bereich von 128 KByte zu adressieren. Aus diesem Grund muß der Prozessor, will er auf alle Bereiche des Bildschirmspeichers zugreifen, im VSDD ein Fenster festlegen, das den gewünschten Adreßbereich freigibt.

Eine 16-Bit-CPU kann direkt zwei Bytes über den Datenbus leiten (Bild 4-70). Mit dem Signal Byte High Enable ($\overline{BHE}$) wird die obere Datenleitung (D_{8-15}) freigegeben. Eine CPU, die über einen 8 Bit breiten Datenbus verfügt, kann nur ein Byte gleichzeitig schreiben oder lesen. Das Low-Byte eines Worts erscheint dabei an den geraden Adressen (2n), das High-Byte des Worts erscheint an den ungeraden Adressen (2n+1). Der Eingang $\overline{BHE}$ des VSDD ist in diesem Fall an +5 V zu legen (Bild 4-72).

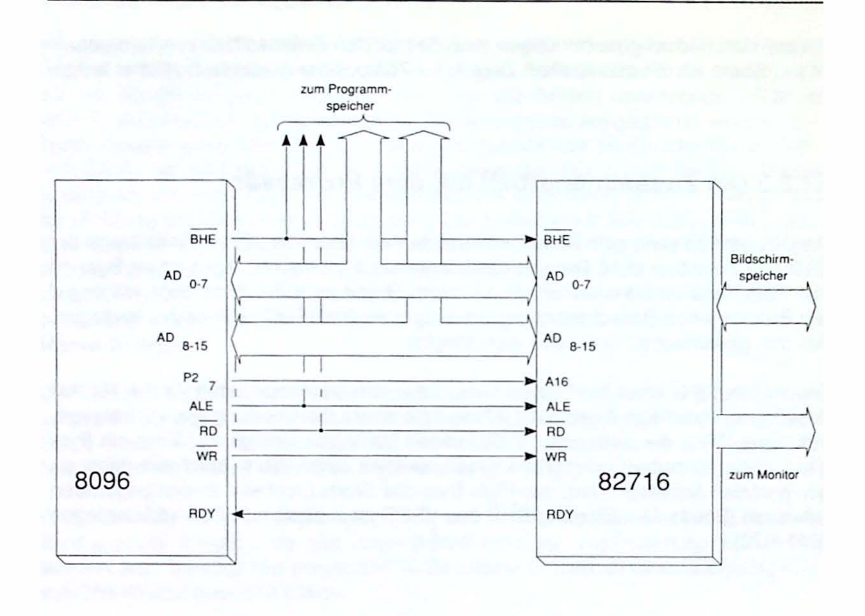


Bild 4-70. Interface des 82716 mit einem 16-Bit-Mikrocontroller

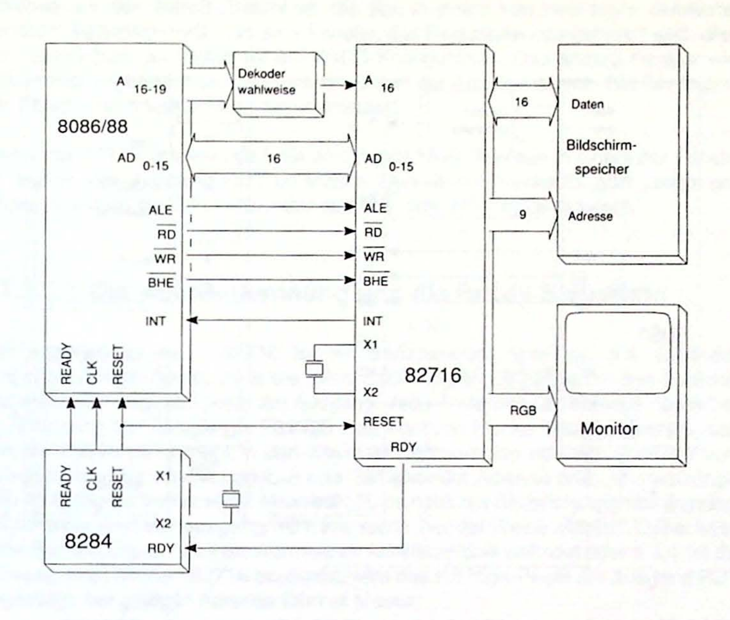


Bild 4-71. Interface des 82716 mit dem 8086/8088

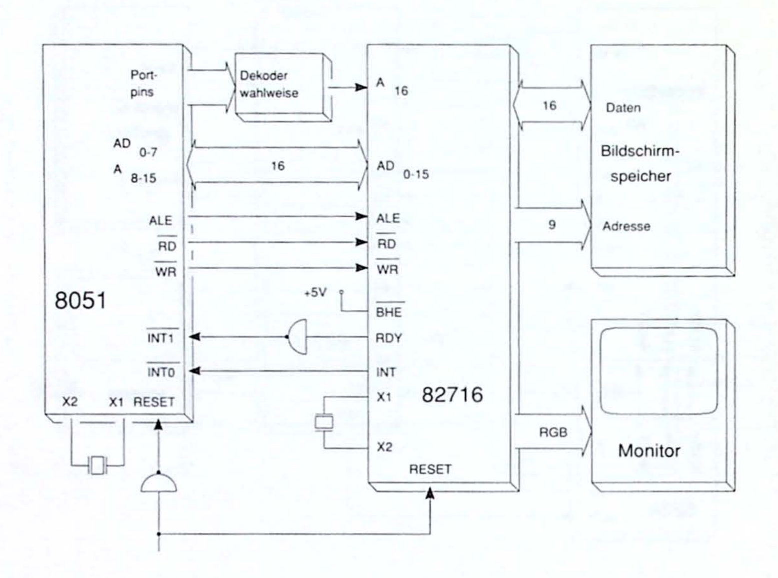


Bild 4-72. Interface des 82716 mit dem 8-Bit-Mikrocontroller 8051/52

Das Ready-Signal des VSDD kann benutzt werden, um Wait-States bei der CPU während eines Bildschirmspeicherzugriffs zu erzeugen. Diese Rückkopplung ist allerdings nicht unbedingt notwendig.

Beim Betrachten des Blockdiagramms (Bild 4-61), stellt man fest, daß der 82716 über keine Adreß- oder Chip-Select-Eingänge verfügt, mit deren Hilfe üblicherweise interne Register zur Programmierung ausgewählt werden. Eine solche Konzeption liegt im 82716 nicht vor. Da er einen sehr großen eigenen Speicher verwalten kann, erübrigt sich die Integration interner Register; er benutzt zu seiner Steuerung vielmehr die Inhalte reservierter Speicherbereiche. Um ihn zu programmieren, braucht man nur diese Speicherbereiche beschreiben. Eine aufwendige Dekodierlogik entfällt dadurch. Der Eingang A_{16} übernimmt die Funktion des $\overline{CS}$-Eingangs. Liegt er an Masse, ist der 82716 selektiert, liegt er an Plus, ist kein Zugriff auf den Bildschirmspeicher über den VSDD möglich.

Wenn die CPU auf den Bildschirmspeicher zugreifen will, legt sie zunächst die Adresse auf den Adreß-/Datenbus, die sich in einem von zwei zuvor definierten Fenstern befinden muß. Das eine Fenster, das Registerfenster genannt wird, dient dem Schreiben der Daten für die VSDD-Konfiguration. Das andere Fenster wird Datenfenster genannt und dient für die Daten der Anzeigeobjekte. Die Festlegung der Fenster wird später ausführlich erläutert.

Wenn der 82716 erkennt, daß die an ihn gerichtete Adresse in einem der Fenster ist, legt er den Ausgang RDY an Masse. Nun kann ein Schreib- oder Lesezugriff erfolgen, indem die CPU entweder das $\overline{WE}$- oder $\overline{RD}$-Signal aktiviert.

4.7.3.3.1 Die Adreßerkennung und die Ready-Steuerung

Der Adreßpuffer des 82716 ist ein transparenter Speicher, d.h. führt das ALE-Signal High-Pegel, fließt die Information an den Eingängen in den Speicher und erscheint augenblicklich am Ausgang. Jede Änderung der Adresse bewirkt eine Änderung der Ausgänge. Erst mit der negativen Flanke des ALE-Signals werden die Daten permanent in den Speicher übernommen und die Eingänge vom Bus getrennt. Die interne Logik, die die Gültigkeit der Adresse prüft, ist unabhängig vom ALE-Signal immer aktiv. Maximal 120 ns nach der Stabilisierung der angelegten Adresse wird der Ausgang RDY in entsprechender Weise aktiviert. Dabei ist es ohne Bedeutung, ob sich die Adresse im Adreßspeicher befindet oder nicht. Ist die Adresse nicht für den 82716 bestimmt, wird das mit High-Pegel am Ausgang RDY angezeigt, bei gültiger Adresse führt er Masse.

4.7.3.3.2 Der Schreibvorgang

Nachdem die Adresse mit dem ALE-Signal in den Baustein geschrieben und dort gespeichert wurde, gibt die CPU nachfolgend die zu schreibenden Daten auf die Datenleitung und setzt das Signal $\overline{WR}$ an Masse. Dabei fließen die Daten in den transparenten Speicher der Bus-Interface-Einheit. Mit der positiven Flanke des $\overline{WR}$-Signals werden die Daten in dem Speicher festgehalten. Wenn die CPU das Ready-Signal (RDY) benutzt, um Wait-States einzufügen, hält sie die $\overline{WR}$-Leitung so lange an Masse, bis der VSDD die RDY-Leitung wieder auf High-Pegel setzt.

Kurze Zeit, nachdem das $\overline{WR}$-Signal aktiviert worden war, transportiert der VSDD die geschriebenen Daten in einen anderen Speicher der CPU. Die Zeitspanne hängt vom Zustand des Systemtakts und von den augenblicklichen Aktivitäten des 82716 ab. Dabei können zwischen zehn und 16 Oszillatorperioden vergehen, bevor der VSDD über die Speicher-Interface-Einheit die Daten in den Bildschirmspei-

cher schreibt. Dieser Schreibvorgang beansprucht nochmals die Zeit von vier oder fünf Perioden je nach Inhalt des SAB-Bits. Bei einer Frequenz von 14,5 MHz resultiert daraus eine Zeitspanne von 0,98 µs bis 1,47 µs.

Eine positive Flanke auf der RDY-Leitung zeigt der CPU an, daß die Daten in die Speicher-Interface-Einheit weitergeleitet wurden. Wenn der Prozessor Wait-States benutzt, kann er nun mit seinen eigenen Aktivitäten fortfahren oder mit dem nächsten Schreibzyklus beginnen.

Wenn das Ready-Signal nicht benutzt wird, geht die $\overline{WR}$-Leitung des Prozessors in aller Regel vor der Ready-Leitung an Plus. Da der 82716 dieselben Schritte durchläuft und keine Rückkopplung besteht, sollte die CPU eine Mindestzeit von 18 Perioden und 100 ns (1,36 µs bei 14,5 MHz) vor dem folgenden Schreibbefehl vergehen lassen.

4.7.3.3.3 Der Lesevorgang

Nach dem Speichern der Adresse mit dem ALE-Signal erkennt der 82716 einen Lesevorgang an der Aktivierung der $\overline{RD}$-Leitung durch die CPU. Der VSDD setzt daraufhin die RDY-Leitung an Masse und gibt augenblicklich die Ausgänge der Bustreiber mit den zu lesenden Daten frei. Das sind allerdings nicht die Daten der gewünschten Adresse, sondern die Daten aus dem vorangegangenen Lesezugriff. Denn der 82716 muß erst einmal selbst die Daten aus dem Bildschirmspeicher lesen (Näheres in Absatz 4.7.3.3.4).

In ähnlicher Weise wie beim Schreiben muß der VSDD zunächst die Adresse an den Bildschirmspeicher richten. Die Minimalzeit der Adressenweiterleitung beträgt dazu acht Perioden, die Maximalzeit 14 Perioden. Das Einlesen der Daten aus dem Bildschirmspeicher benötigt nochmals fünf Perioden.

Nun besteht durch Löschen eines internen Bits, das Pipline-Read-Enable-Bit (PRE), die Möglichkeit, mit Hilfe der Ready-Leitung die CPU solange warten zu lassen, bis die Speicher-Interface-Einheit über die gewünschten Daten verfügt. Ist dieses Bit gelöscht, werden die Daten augenblicklich an die Ausgabepuffer weitergeleitet und die RDY-Leitung auf High-Pegel gesetzt. Die CPU verläßt den Wait-State und übernimmt die an den Ausgängen des 82716 stehenden Daten. Bei der Verwendung von Mikrocontrollern als CPU sollten die Signalleitungen $\overline{PSEN}$ und $\overline{RD}$ nicht zusammengefaßt sein.

4.7.3.3.4 Das Lesen von Daten im Pipeline-Modus

Wenn das Pipline-Read-Enable-Bit (PRE) gesetzt ist, ist der Pipeline-Modus für das Lesen von Daten gewählt. Diese Möglichkeit ist für alle Prozessoren vorgesehen, die über keinen Ready-Eingang verfügen (Mikrocontroller) und für solche Anwendungen, bei denen große Datenmengen schnell gelesen werden sollen. Im Pipeline-Modus werden immer die Daten der vorausgegangenen Adresse gelesen. Der Inhalt des ersten Lesezugriffs ist zu verwerfen. Sollen n Daten gelesen werden, müssen n+1 Lesezyklen ausgeführt werden. Die Adresse des letzten Lesebefehls kann beliebig sein. Soll der Inhalt von nur einer Adresse gelesen werden, sind somit zwei Zugriffe auf den 82716 notwendig.

Beim Schreiben ist dieser Modus nicht notwendig, da die CPU die Daten in den VSDD schreibt und nicht warten muß, bis der Baustein sie an die richtige Stelle transportiert hat. Das ist beim Lesen anders. Hier kann die CPU erst fortfahren, wenn tatsächlich die Daten bei ihr eingetroffen sind. Der Pipline-Modus umgeht unnötige Wartezyklen. Durch das Setzen des PRE-Bits wird auch in Zusammenarbeit mit einer langsamen CPU verhindert, daß die Daten augenblicklich von der Speicher-Interface-Einheit zur Bus-Interface-Einheit transportiert werden. Dieser Vorgang läuft erst zu Beginn des folgenden Lesezugriffs ab. Ein Vergleich des Pipeline-Modus mit dem Ready-Modus gibt Bild 4-73. Auch hier beträgt die Zeit von Beginn der Aktivierung des $\overline{RD}$-Signals bis zur positiven Flanke das ALE-Signals 18 Perioden plus 100 ns.

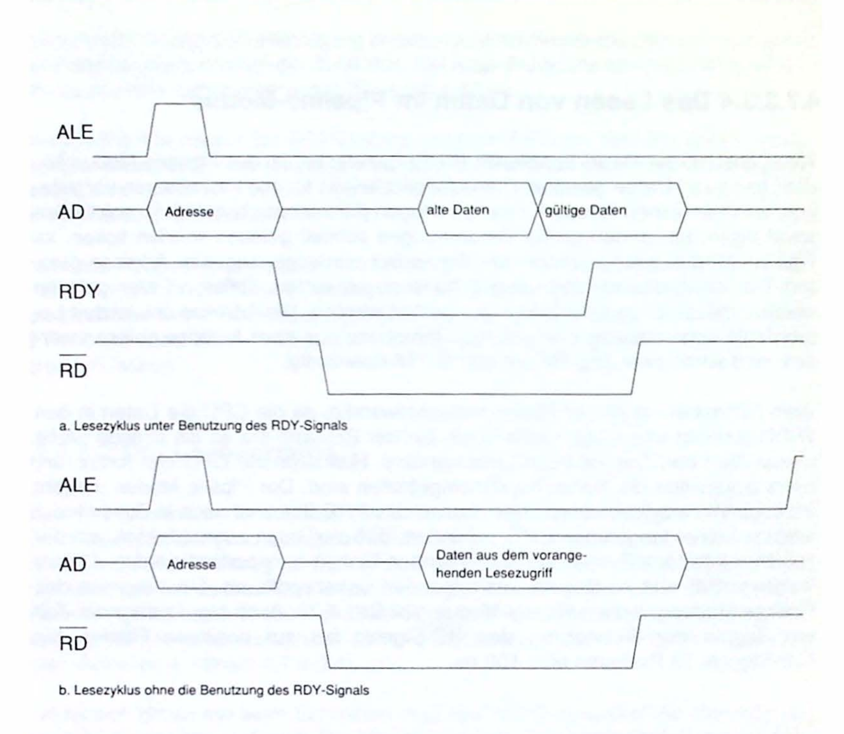


Bild 4-73. Signalformen bei einem Lesezugriff auf den 82716

4.7.3.3.5 Der freie Zugriff

Wenn der Ready-Pin zur Erzeugung von Wait-States nicht benötigt wird, kann man den 82716 so programmieren, daß er der CPU anzeigt, wann sich der VSDD zwischen zwei Bildaufbauten befindet. In dieser Zeit ist der Zugriff der CPU auf den Bildschirmspeicher nicht durch den Eigenbedarf des VSDD für die Zeilenkonstruktion eingeschränkt. Während dieser Zeit hat die CPU mehr oder weniger "freien Zugriff" auf das DRAM.

Um dem RDY-Pin die zweite Funktion zuzuweisen, muß das Free-Access-Enable-Bit (FAE) gesetzt werden. Dieses Bit ist ebenso wie das Pipeline-Bit Bestandteil des Video Konfigurationsregisters 1 (R1).

4.7.3.3.6 Die Einschränkung des Zugriffs

Natürlich kann die CPU zu jeder Zeit auf das DRAM zugreifen und diese Zugriffe haben immer Priorität vor allen anderen Aktivitäten des VSDD, auch vor der Zeilenkonstruktion. Zu viele Zugriffe können dabei bewirken, daß die Zeile unvollständig an den Monitor abgegeben wird, denn die Zeit für eine Konstruktion kann nicht gedehnt werden. Dieser Sachverhalt wird der CPU durch High-Pegel am Interrupt-Pin (INT) angezeigt.

Nun ist es aber auch möglich, bei Prozessoren, die über einen funktionierenden Ready-Eingang verfügen, Wait-States zu erzeugen, wenn bei der Zeilenkonstruktion zu viele Zugriffe erfolgen und die Zeile unvollständig an den Monitor gesendet würde. Der VSDD kann so programmiert werden, daß er nur eine Mindestzahl an Zugriffen während der Zeilenkonstruktion zuläßt. Ist diese Zahl überschritten, gibt er den Ausgang RDY an Masse und stoppt so die Aktivitäten der allzu eifrigen CPU. Es können n Zugriffe programmiert werden, wobei gilt $n \leq 16$. Bei einem Zugriff der Nummer n+1 geht der Ausgang RDY solange an Masse, bis die aktuelle Zeile fertiggestellt ist.

Diese Option wird durch Setzen des Bits Priority Counter Enable (PCE) in Register 1 gewählt. Der Wert für n wird in vier Bits des Registers 6 geschrieben.

4.7.3.4 Der Synchronisierungsgenerator

Der Synchronisierungsgenerator ist eine Einheit zur Zeittaksteuerung, dessen Hauptaufgabe darin besteht, die Synchronisierungssignale für den Bildschirm und die internen Interrupts bei beendeter Zeilenkonstruktion sowie fertigem Bildaufbau zu erzeugen. Damit ein sehenswertes Bild auf dem Monitor erscheint, müssen die Pixel-Informationen in geordneter Weise, d.h. mit derselben Zeitkonstanz an den Bildschirm gesendet werden. Der VSDD unterteilt den Bildschirm in eine aktive Zone, in der alle Pixel erscheinen, und in einen Rahmen, auf dem keine Informationen zu finden sind (Bild 4-74). Damit eine Bildschirmzeile genau unter der anderen zu stehen kommt, müssen vier horizontale Zeitparameter programmiert werden:

1. Pulsbreite t_0 des Synchronisierungssignals
   Bezeichnung: HC0 (HC = Horizontal Constant)
2. Start t_1 der aktiven Anzeigenzone
   Bezeichnung: HC1
3. Ende t_2 der aktiven Anzeigenzone
   Bezeichnung: HC2
4. Zeilenbreite t_3
   Bezeichnung: HC3

Die einzelnen Zeiten finden sich im Diagramm von Bild 4-74.

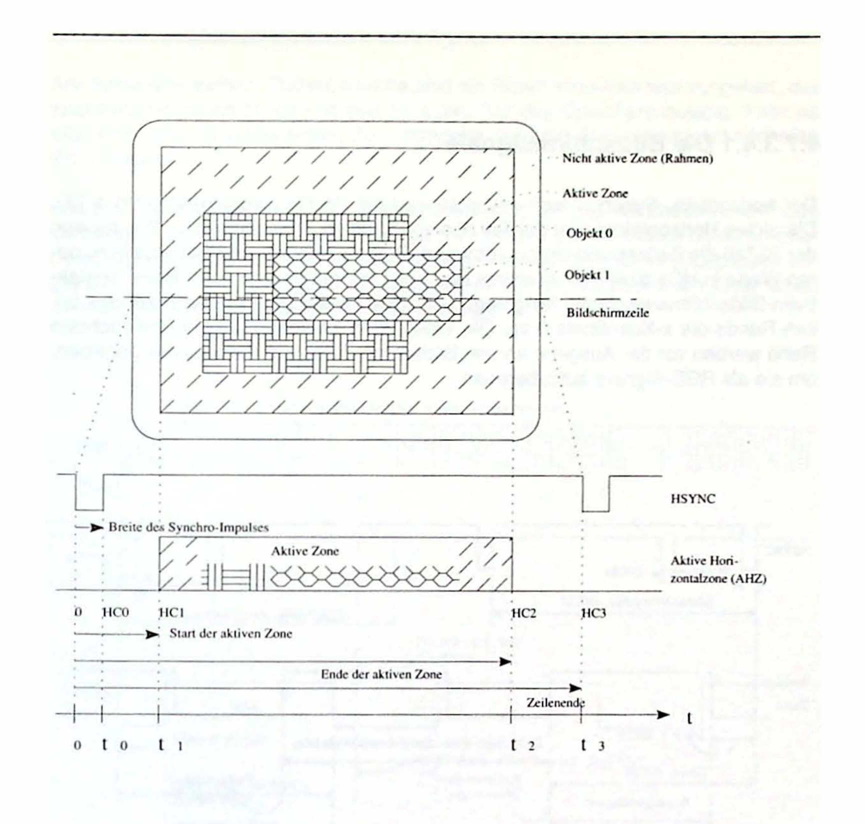


Bild 4-74. Die aktive Bildschirmzone und die programmierbaren Rasterparameter

In ähnlicher Weise wie die horizontale Synchronisation muß auch die vertikale Synchronisation vorgenommen werden. Es muß dem Monitor mitgeteilt werden, wann der neue Seitenaufbau beginnt, wann die erste bzw. die letzte Zeile auftritt und wann der Bildaufbau zu Ende ist. Die Pulsbreite des vertikalen Synchronisierungssignals trägt die Bezeichnung VC0, der Start der ersten Zeile VC1, das Ende des aktiven Bereichs VC2 und die Zeit bis zum vollständigen Aufbau des Bildschirms ist in VC3 abgelegt. Die Bits HCX und VCX sind Inhalte der Register R12 bis R15.

4.7.3.4.1 Die Bildschirmsignale

Der horizontale Synchronisationsimpuls wird mit HSYNC abgekürzt (Bild 4-75). Die aktive Horizontalzone ist der Teil des horizontalen Bildschirmbereichs, für den der 82716 die Zeilenkonstruktion vornimmt. Durch die Start- und Stoppzeiten, deren Werte in HC1 bzw. HC2 abgelegt sind, wird der rechte und linke Rand des aktiven Bildschirmausschnitts festgelegt. Der VSDD weist dem ersten Pixel des linken Rands die x-Koordinate 0 zu. Die Videodaten zwischen linkem und rechtem Rand werden vor der Ausgabe an den Bildschirm durch die Farbpalette gegeben, um sie als RGB-Signale aufzubereiten.

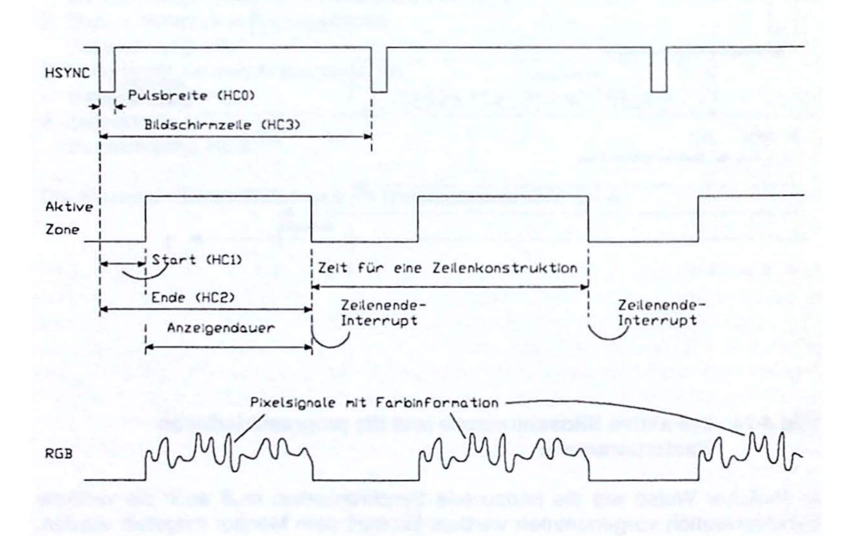


Bild 4-75. Die Signale für die horizontale Steuerung

Am Ende des aktiven Zeilenbereichs wird ein Zeilenende-Interrupt ausgelöst, der nachfolgend einen Block-Refresh für einen Teil des Speichers auslöst. Falls es sich nicht um die letzte aktive Zeile handelte, wird mit dem Aufbau der nächsten Zeile begonnen.

Wie bereits erwähnt, muß auch eine senkrechte Steuerung des Bildschirmaufbaus vorgenommen werden. Durch die negativen Flanke des vertikalen Synchronisierungssignals wird dem Monitor der Neubeginn eines kompletten Bildaufbaus angezeigt (Bild 4-76). Die Start und Stoppzeiten der vertikalen aktiven Zone legen den oberen und den unteren Rand der Anzeige fest.

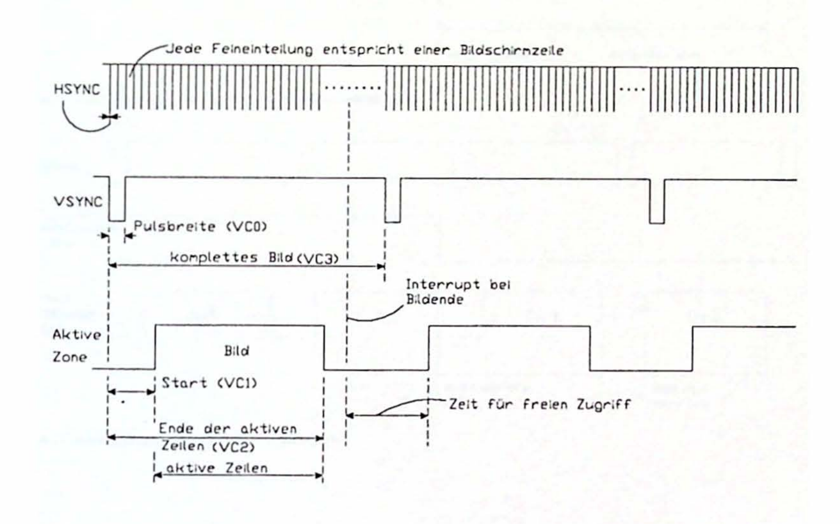


Bild 4-76. Die Signale für die vertikale Steuerung

Zu beachten sind die Zeiten und Signale bei Beginn der vertikalen aktiven Zone (Bild 4-77.a). Die Konstruktion der ersten Zeile beginnt am Ende der ersten aktiven horizontalen Zone, die auf die positive Flanke der aktiven vertikalen Zone folgt. Dieser Zeilenaufbau kann bis zum Ende der zweiten aktiven vertikalen Zone dauern, wobei der Koordinierungsblock über einen Zeilenstopp-Interrupt die Aufforderung zur Konstruktion der zweiten Zeile erhält. Die erste aktive Zeile wird somit also erst in der dritten aktiven Zone auf dem Bildschirm angezeigt.

In ähnlicher Weise hinkt die Anzeige am unteren Rand des aktiven Bildschirms dem Signal zur Beendigung der aktiven vertikalen Zone hinterher (Bild 4-77.b). Mit der Konstruktion der letzten Zeile wird am Ende der aktiven Zone n begonnen. Sie wird in der Zone n+2 am Bildschirm angezeigt.

Am unteren Ende des aktiven Bildschirmausschnitts, nach der Konstruktion der letzten Zeile, wird ein Interrupt erzeugt, der dem Koordinierungsblock die Beendigung des Bildaufbaus anzeigt. Dabei werden im VSDD verschiedene Adressen und interne Flip-Flops aktualisiert. Diese Sequenz benötigt eine Zeit von 740 Oszillatorperioden; das sind 51 µs bei 14,5 MHz.

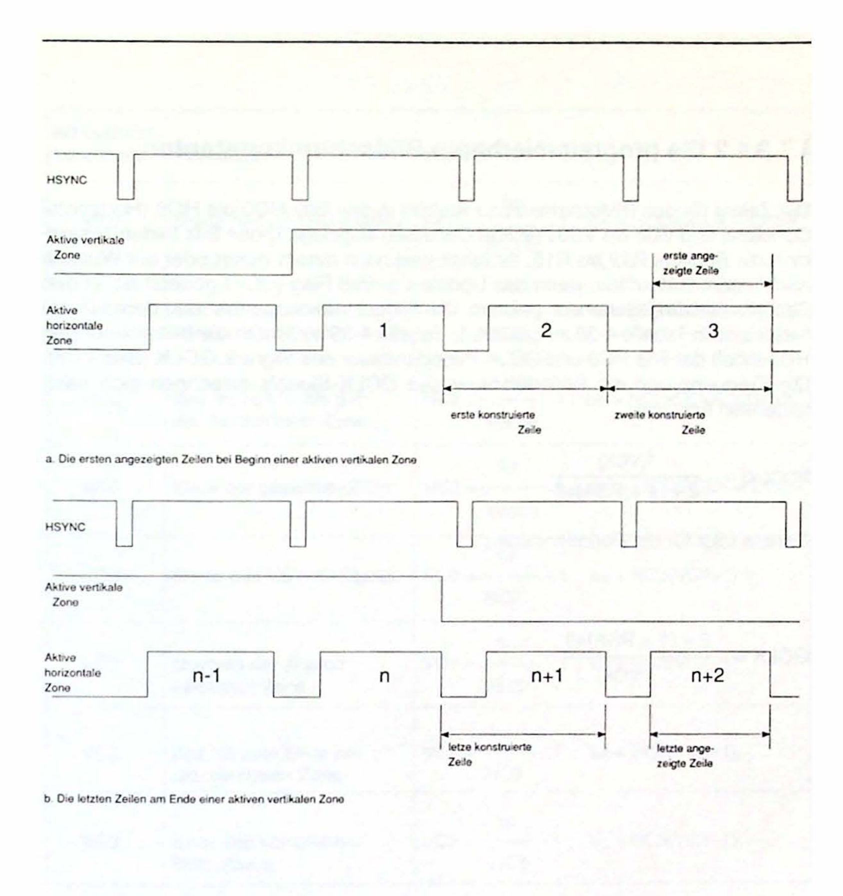


Bild 4-77. Die Zeilenkonstruktion am oberen und unteren Rand des aktiven Bildschirms

4.7.3.4.2 Die programmierbaren Bildschirmkonstanten

Die Zeiten für den Bildschirmaufbau werden in den Bits HC0 bis HC3 (Horizontal Constant) und VC0 bis VC3 (Vertical Constant) abgelegt. Diese Bits bilden zusammen die Register R12 bis R15. Ihr Inhalt wird nach einem Reset oder auf Wunsch nach jedem Bildaufbau, wenn das Update Controll Flag (UCF) gesetzt ist, in den Synchronisierungsgenerator gelesen. Die Bildschirmkonstanten und deren Parameter sind in Tabelle 4-39 aufgeführt. In Tabelle 4-39 bedeuten die Bezeichnungen HC0 Inhalt der Bits HC0 und GCLK Periodendauer des Signals GCLK (Bild 4-78). Die Frequenz und die Periodendauer des GCLK-Signals errechnen sich nach folgenden Formeln:

$$f_{(GCLK)} = \frac{f_{(VCK)}}{2 \bullet (1 + PSA) \bullet 8}$$

Daraus folgt für die Periodendauer:

$$GCLK = \frac{2 \bullet (1 + PSA) \bullet 8}{f_{(VCK)}}$$

Bildschirm-konstante	Parameter	Formel
HC0	Breite des HSYNC-Signals	$HC0 = \frac{t_{h0}}{t_{(GCLK)}} - 1$; $t_{h0} = HC0 \bullet GCLK + GCLK$
HC1	Startzeit der aktiven horizontalen Zone	$HC1 = \frac{t_{h1}}{t_{(GCLK)}} - 1$; $t_{h1} = HC1 \bullet GCLK + GCLK$
HC2	Zeit bis zum Ende der akt. horizontalen Zone	$HC2 = \frac{t_{h2}}{t_{(GCLK)}} - 1$; $t_{h2} = HC2 \bullet GCLK + GCLK$
HC3	Ende der gesamten Zeile	$HC3 = \frac{t_{h3}}{t_{(GCLK)}} - 1$; $t_{h3} = HC3 \bullet GCLK + GCLK$
VC0	Breite des VSYNC-Signals	$VC0 = \frac{t_{v0}}{HC3} - 1$; $t_{v0} = HC3(VC0+1)$
VC1	Startzeit der aktiven vertikalen Zone	$VC1 = \frac{t_{v1}}{HC3} - 1$; $t_{v1} = HC3(VC1+1)$
VC2	Zeit bis zum Ende der akt. vertikalen Zone	$VC2 = \frac{t_{v2}}{HC3} - 1$; $t_{v2} = HC3(VC2+1)$
VC3	Ende des kompletten Bildaufbaus	$VC3 = \frac{t_{v3}}{HC3} - 1$; $t_{v3} = HC3(VC3+1)$

Tabelle 4-39. Bildschirmkonstanten und Berechnungsformeln

Die horizontalen Bildschirmkonstanten HC0 bis HC3 werden als Maßzahl der GCLK-Perioden minus eins programmiert, d.h. hat der Inhalt von HC0 den Wert 0, ist die Dauer des HSYNC-Signals eine GCLK-Periode lang. Hat die Konstante den Wert 1, beträgt die Dauer zwei usw.

Das GCLK-Signal stammt aus dem Videotaktsignal (VCK) wie in Bild 4-78 gezeigt. Der Synchronisierungsgenerator wird durch das VPH-Signal getaktet, dessen Frequenz entweder die Hälfte oder ein Viertel der Videofrequenz ist. Die höchstzulässige VPH-Frequenz beträgt 8 MHz. Daher sollte das PSA-Bit bei einem Videotakt von mehr als 16 MHz gesetzt sein. Die Frequenz des GCLK-Signals ist immer ein Achtel der VPH-Frequenz. Die Periodendauer des GCLK-Signals legt die Einheit der horizontalen Zeitkonstanten fest.

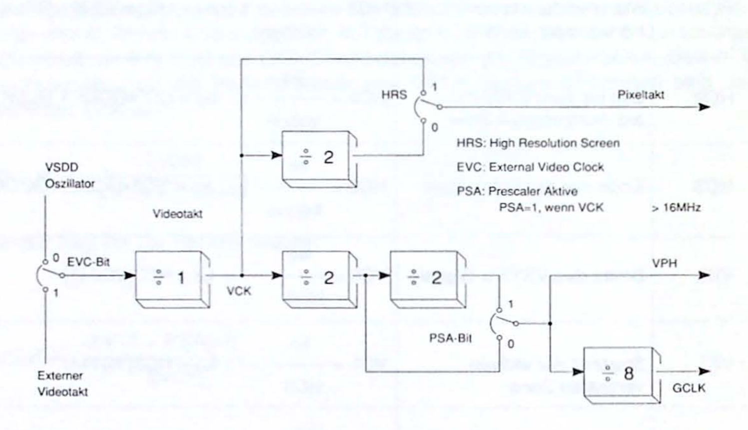


Bild 4-78. Die Takterzeugung für den Bildaufbau

Ein Beispiel:

Beträgt der Videotakt 20 MHz, muß das PSA-Bit gesetzt werden, so daß die VPH-Frequenz 5 MHz ist. Die GCLK-Frequenz von 625 kHz legt die Einheit der horizontalen Zeitkonstanten auf 1,6 µs fest. Der Pixel-Takt in Bild 4-78 kann entweder von derselben oder der halben Frequenz des Videotakts sein. Somit können in Abhängigkeit der Bits HSR und PSA 8, 16 oder 32 Pixel pro GCLK-Periode auftreten.

Der Bit-Bereich zur Programmierung der Horizontalkonstanten HC0 bis HC3 umfaßt in den Registern R12 bis R15 jeweils 6 Bits. Deswegen kann die Maximalzahl den Wert 63 nicht überschreiten, d.h. daß die Gesamtzahl pro Zeile höchstens 64 GCLK-Perioden beträgt. Somit liegen die Zeiten für eine Zeile bei einem 70 ns-Takt und PSA=0 zwischen 1,12 µs und 71,68 µs in Schritten von 1,12 µs. Steht der Wert 56 in HC3, resultiert daraus eine Horizontalperiode von 57•1,12 µs = 63,84 µs.

Die interne Hardware stellt folgende Anforderungen an die horizontalen Bildschirmkonstanten:

HC0 < HC1 < ½ HC3 < HC2 < HC3

Die senkrechten Bildschirmkonstanten VC0 bis VC3 werden in Einheiten von horizontalen Zeilen programmiert. Ist der Inhalt 0, folgt daraus die Zeit von einer horizontaler Zeile. Der Bit-Bereich zur Programmierung der Vertikalkonstanten VC0 bis VC3 umfaßt in den Registern R12 bis R15 jeweils 10 Bits. Somit können Werte bis zum Maximalwert von 1023 programmiert werden. Es sind also 1024 Zeilen auf dem Bildschirm darstellbar. Der Inhalt von VC3 = 261 erzeugt 262-Zeilen-Feld.

Die interne Hardware stellt folgende Anforderungen an die vertikalen Bildschirmkonstanten:

VC0 < VC1 < VC2 < VC3

Das folgende Beispiel erläutert die Berechnung der Werte für einen IBM CGA-Monitor:

Voraussetzungen:
- Der Viedotakt beträgt 14,5 MHz
- PSA=0, EVC=0, HRS=1
- Der Pixel-Takt ist 14,5 MHz
- Der GCLK-Takt beträgt 14,5:16 = 0,90625 MHz; die GCLK-Periode ist 1,1 µs.

Die Forderungen des CGA-Monitors an die Horizontalzeiten:
- Breite des Zeilenstartimpulses (HSYNC): 2,2 µs
- Horizontale Startzeit: 11,0 µs
- Horizontale Stoppzeit: 55,0 µs
- Gesamtzeit der Zeile: 64,0 µs

Die zu programmierenden horizontalen Werte betragen:
- HC0 = 2,2:1,1 - 1 = 1 = 000001b
- HC1 = 11,0:1,1 - 1 = 9 = 001001b
- HC2 = 55,0:1,1 - 1 = 49 = 110001b
- HC3 = 64,0:1,1 - 1 = 57 = 111001b

Die Forderungen des CGA-Monitors an die Vertikalzeiten:
- Startimpuls für neuen Bildaufbau: 0,26 ms
- Vertikale Startzeit: 2,05 ms
- Vertikale Stoppzeit: 14,85 ms
- Zeit für den gesamten Bildschirm: 16,00 ms

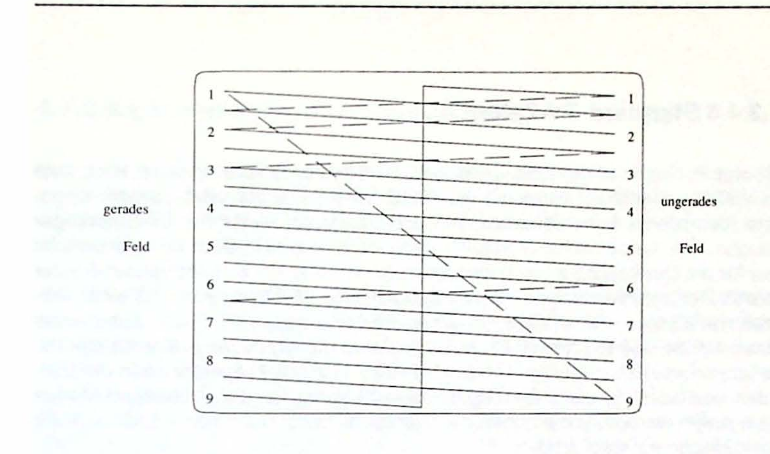
Die zu programmierenden vertikalen Werte betragen:
- VC0 = 256:64 - 1 = 3 = 0000000011_b
- VC1 = 2050:64 - 1 = 31 = 0000011111_b
- VC2 = 14850:64 - 1 = 231 = 0011100111_b
- VC3 = 16000:64 - 1 = 249 = 0011111001_b

In die Register R12 bis R15 sind folgenden Werte zu schreiben:
- R12 = 0000010000000011_b = 0403_h
- R13 = 0010010000011111_b = $241F_h$
- R14 = 1100010011100111_b = $C4E7_h$
- R15 = 1110010011111001_b = $E4F9_h$

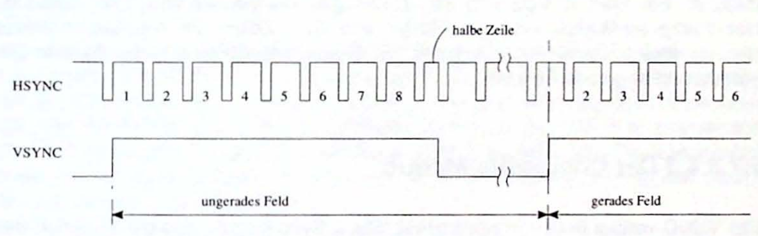
Es sollte nicht vergessen werden das Register R3 mit dem Wert 0140_h = 640_d zu beschreiben und das Bit DEI auf 1 zu setzen, um die Ausgänge dem IBM-Farb-Monitor anzupassen.

4.7.3.4.3 Der Interlaced-Modus

Wenn das INL-Bit in Register 1 gesetzt ist, ändert der VSDD seine Synchronisierungssignale in der Art, daß ein ineinander verflochtenes Anzeigenraster entsteht. In diesem Modus besteht der Bildschirm aus zwei Feldern: ein gerades und ein ungerades Feld. (Der VSDD zeigt dieselbe Information auf beiden Feldern an.) Der vertikale Synchronisationsimpuls für das gerade Feld wird um ein halbe Zeilenlänge verzögert, so daß die Zeilen des geraden Felds zwischen die Zeilen des ungeraden Felds zu liegen kommen. Die Wirkung wird in Bild 4-79 gezeigt. Es sind die horizontalen und vertikalen Synchronisierungsimpulse für ein 17-Zeilen-Interlaced-Raster gezeigt. Bild 4-79 a zeigt die Raster, wie sie auf dem Bildschirm erscheinen werden. Die Rückführung des Elektronenstrahls ist in punktierten Linien gehalten. Um die Übersichtlichkeit nicht zu gefährden sind nur wenige Strahlrückführungen gezeichnet.



a. Die Anzeige von 17 Zeilen im Interlaced-Modus



b. Die Zeittaktsteuerung für ein 17-Zeilen-Interlaced-Raster

Bild 4-79. Die Anzeige im Interlaced-Modus

Die Zeile 9 des geraden Feldes in Bild 4-79 wird exakt in der Mitte unterbrochen. Wenn die Unterbrechung an anderer Stelle der Zeile erfolgt, resultiert daraus ein ungleicher Zeilenabstand. Damit der VSDD die Mitte genau treffen kann, muß die Zeilenlänge eine gerade Zahl von GCLK-Perioden aufweisen, d.h. die Bildschirmkonstante HC3 muß einen ungeraden Inhalt haben.

4.7.3.4.4 Standard-TV-Zeiten

Die europäischen und die amerikanischen TV-Standards können ohne Probleme vom VSDD gehandhabt werden. Der VSDD kann separate oder zusammengesetzte (composite) Synchronisierungen und Interlaced-Bilder für TV-Empfänger erzeugen. Der europäische Standard verlangt eine 50-Hz-Bildrate und benutzt 64 µs für die Darstellung einer Bildschirmzeile, woraus ein 625-Interlaced-Raster entsteht. Der amerikanische Standard benutzt eine 60-Hz-Bildrate mit einer Zeilenzeit von 63,6 µs, der ein 525-Interlaced-Raster erzeugt. Die Benutzung eines preiswerten 14,5-MHz-Farb-TV-Quarzes gestattet die Erzeugung aller für die TV-Systeme erforderlichen Zeiten. Mit dem VP-Bit im Registersegment kann die Länge des vertikalen Synchronisierungsimpulses korrekt für den Interlaced-Modus programmiert werden. Für europäische Standards beschreibt man es mit Null, für amerikanische mit einer Eins.

Beispiel: Ein Wert in VC3 von 311 Zeilen gibt ein Raster von 312 Zeilen im Non-Interlaced-Modus, und ein Raster von 625 Zeilen im Interlaced-Modus (INL=1). Im Interlaced-Modus fügt der VSDD automatisch eine halbe Zeile in das gerade und ungerade Feld ein.

4.7.3.4.5 Der Composite-Modus

Der VSDD verfügt über ein Kontroll-Bit (SM = Sync Mode), das die Funktion des Pins HSYNC ändern kann. Ist es gelöscht (SM = 0), geben die Ausgänge HSYNC und VSYNC wie üblich die HSYNC- und VSYNC-Signale aus. Ist es gesetzt (SM = 1), gibt der Pin VSYNC das VSYNC-Signal aus, der HSYNC-Pin gibt aber ein Signal aus, das durch eine Exklusiv-Oder-Verknüpfung der Signale HSYNC und VSYNC entstanden ist und das als Composite-SYNC-Signal verwendet werden kann.

4.7.3.4.6 Die externe Synchronisation

Der Videobereich des VSDD kann auf ein externes Videosignal nach zwei Arten programmiert werden: er kann mittels integrierter Phase-Locked-Loop-Schaltung (PLL) die eigene Phase auf ein externes Signal synchronisieren oder aber er kann im Zwillings-Modus arbeiten.

Das Kontroll-Bit MAS (Master) gestattet die Wahl zwischen interner und externer Synchronisation. Bei gesetztem Bit erzeugt der VSDD Synchronisationsimpulse und gibt sie an den HSYNC- und VSYNC-Pins aus. Bei gelöschtem Bit wartet der VSDD auf extern erzeugte Synchronisationsimpulse. Wenn die externen Synchronisierungssignale getrennt vorliegen, werden sie an die entsprechenden Pins (HSYNC, VSYNC), die in diesem Fall als Eingänge konfiguriert sind (MAS = 0; SM = 0), angelegt. Liegen sie als Composite-Signal vor (MAS = 0; SM = 1), wird das Composite-Signal dem Eingang HSYNC zugeführt; der VSYNC-Pin gibt das VSYNC-Signal aus.

Um die interne PLL-Schaltung zu benutzen, muß das MAS-Bit in Register 1 gelöscht sein. Der VSDD muß dabei so programmiert sein, daß er sich in weitestgehender Übereinstimmung mit dem externen Signal befindet. Das betrifft vor allem die Zahl der Bildschirmzeilen, die vertikale Pulsbreite und den Interlaced-Modus. Der Taktgenerator der 82716 muß dabei in der Lage sein, den Frequenzbereich soweit zu variieren, so daß die komplette Zeit für eine Zeile (HC3) exakt die gleiche wie die des externen Signals ist.

Sind diese Anforderungen erfüllt, wird die externe Synchronisation auf zwei Stufen ausgeführt: Die Bildschirmstufe und die Pixel-Stufe.

In der Bildschirmstufe bewirkt eine negative Flanke im externen VSYNC-Signal das Rücksetzen des VSDD auf den Beginn des eigenen Bildschirmfelds. Dadurch werden externes Signal und VSDD-Signal auf den Bildschirmanfang synchronisiert.

Auf der Pixel-Stufe bewirkt eine negative Flanke des externen HSYNC-Signals eine Überprüfung des VSDD-internen HSYNC-Signals, um zu bestimmen, ob sein eigener Videotakt im Vergleich zum externen Signal zu schnell oder zu langsam ist. Ist bei dieser externen negativen Flanke das interne HSYNC-Signal bereits an Masse, wird der interne Videotakt als zu schnell angesehen, und am Ausgang PLLCTL erscheint eine Null. Ist andererseits das HSYNC noch nicht an Masse, heißt das, der Videotakt ist zu langsam, und der Pin PLLCTL gibt eine Eins aus.

Mit diesem binären Signal kann der Oszillator zur Beschleunigung oder Verlangsamung veranlaßt werden, was zu einer Synchronisierung des externen mit dem internen Videosignal führt.

Die Frequenz eines Schwingkreises auf Quarz-Basis kann damit nur in einem Bereich von ±0,03 Prozent seiner Fundamentalfrequenz geändert werden.

Der Ausgang PLLCTL kann so gesteuert werden, daß er entweder die ganze Zeit oder nur für die Zeit des horizontalen Impulses aktiv ist. Sonst ist er im Tri-State. Er wird mit dem Bit PLL in Register R2 zum Leben erweckt. Die zweite Methode ist empfehlenswert, wenn ein LC-Oszillator benutzt wird.

4.7.3.4.7 Die Synchronisation im Zwillings-Modus

Die Synchronisation im Zwillings-Modus hängt vom Zustand der Bits TwinMode Master (TMM) und TwinMode Slave (TMS) in Register R1 ab. Sind beide Bits Null, wird der Zwillings-Modus nicht benutzt. Für den Master muß gelten: TMM = 1 und TMS = 0; für den Slave TMM = 0 und TMS = 1. Der Zustand TMM = 1 und TMS = 1 ist nicht zulässig.

Der Master erzeugt im Zwillings-Modus die Synchronisierungssignale in gewohnter Weise. Der VSYNC-Pin des Masters wird dabei lediglich mit dem Eingang HSYNC des Slaves verbunden. Mit den vertikalen Synchronisationssignalen des Masters wird der Slave synchronisiert.

Tabelle 4-40 faßt alle Synchronisierungs-Modi zusammen.

TMM	TMS	SM	MAS	HSYNC	VSYNC	PLLCTL
0	0	0	0	HSYNC ein	VSYNC in	CT L aus
0	0	0	1	HSYNC aus	VSYNC aus	T
0	0	1	0	CSYNC ein	VSYNC aus	CTL aus
0	0	1	1	CSYNC aus	VSYNC aus	T
0	1	0	0	RESET ein	VSYNC aus	T
0	1	0	1	RESET ein	VSYNC aus	T
0	1	1	0	RESET ein	VSYNC aus	T
0	1	1	1	RESET ein	VSYNC aus	T
1	0	0	0	HSYNC ein	RESET aus	CTL aus
1	0	0	1	HSYNC aus	RESET aus	T
1	0	1	0	CSYNC ein	RESET aus	CTL aus
1	0	1	1	CSYNC aus	RESET aus	T

Legende:
ein : Pin als Eingang konfiguriert
aus : Pin als Ausgang konfiguriert
CSYNC : Composite-Signal
HSYNC : horizontales Synchronisierungssignal
VSYNC : vertikales Synchronisierungssignal
CTL : PLL-Signal
RESET : Reset für den Slave im Zwillings-Modus
T : Tri-State

Tabelle 4-40. Die Synchronisations-Modi des 82716

4.7.3.5 Die Pixel-Einheit

Die Pixel-Einheit ist eine Gruppe von verschiedenen Blöcken im 82716, die die digitalen Anzeigedaten in RGB-Signale umwandelt. Sie besteht aus dem Formatierer, zwei Zeilenspeicher, der Farbtabelle und dem RGB-Digital-Analog-Konverter (DAC) (Bild 4-80).

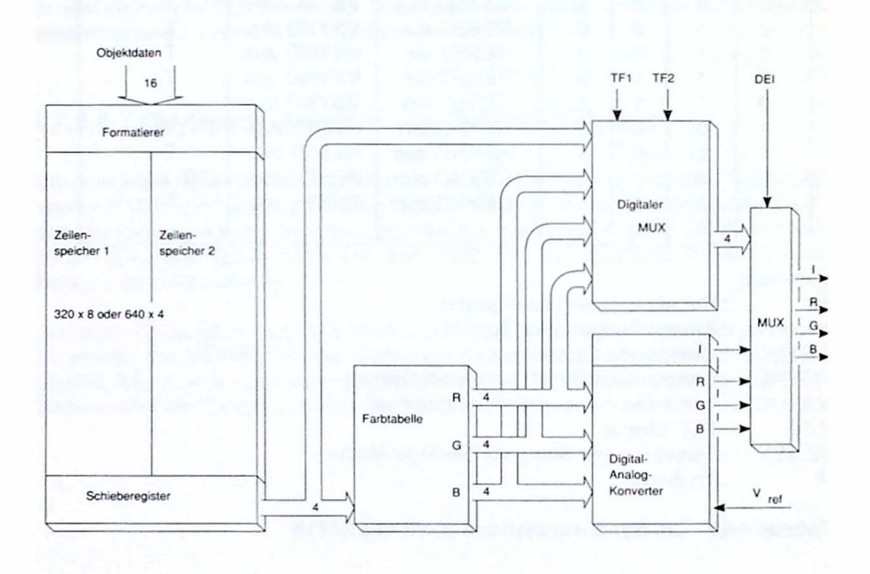


Bild 4-80. Die Funktionsblöcke der Pixel-Einheit

Der Formatierer erhält die rohen Daten aus dem Bildschirmspeicher und formt sie mit den Daten aus dem Videokonfigurationsregister, der Objekt-Beschreibungstabelle und dem Charakter-Generator zu einer vollständigen Zeile um. Die Zeilenspeicher sind dynamische RAM-Zellen, die sich im 82716 befinden. Jeder Zeilenspeicher besteht aus vierzig 64-Bit-Words.

4.7.3.5.1 Die Zahl der Pixel pro Zeile

Man kann den 82716 so programmieren, daß er pro Zeile entweder 640 Pixel mit vier Bit je Pixel oder 320 Pixel mit acht Bit je Pixel aufnimmt.

Für eine Anzeige, die mehr als 320 Pixel in einer Zeile verlangt, können pro Pixel nur vier Bits zur Verfügung gestellt werden, und der VSDD befindet sich im hochauflösenden Modus. Mit diesen vier Pixel wird eine Farbe aus 16 verschiedenen Kombinationen der Farbtabelle gewählt. Dieser Modus wird eingestellt, indem man das Bit HSR (High Screen Resolution) im Register R0 auf Eins setzt.

Mehr Auswahl in der Farbe hat man durch die Verwendung von acht Bits je Pixel. Dabei muß man auf die Farbtabelle im Bildschirmspeicher und auf externe Digital-Analog-Konverter (DAC) zurückgreifen. Um diesen bunteren Modus zu wählen, muß das HSR-Bit gelöscht werden. Damit der VSDD die interne Farbtabelle und den integrierten Digital-Analog-Konverter ignoriert, muß das DEI-Bit (Digitally Encoded Information) gesetzt sein. Der 82716 gibt dann die unveränderte digitale Information gemultiplext zu je vier Bits an den Ausgängen R, G, B und I aus. Um die parallele 8-Bit-Information wieder zu erhalten, müssen die Daten extern demultiplext werden. Das kann mit Hilfe des Takts am Ausgang CKIO erfolgen.

Die übliche Auflösung eines Farbmonitors benutzt 320 Pixel pro Zeile. Sie erhält man, indem man sowohl das HER- als auch das DEI-Bit löscht. Mit HSR = 0 benutzt jedes Pixel acht Bits im Zeilenspeicher (aber nicht notwendigerweise auch im externen Bildschirmspeicher). Das Löschen des DEI-Bits schaltet die interne Farbtabelle und den internen Digital-Analog-Konverter ein, der aber nur die unteren vier Bits eines Pixels benutzt.

Die maximale Zeilenlänge beträgt also 320 bzw. 640 Pixel. Die tatsächlich dargestellte Zahl an Pixel kann mit dem Produkt aus der Dauer der aktiven horizontalen Zone und der Pixel-Rate errechnet werden. Die Pixel-Rate ist die Frequenz des Videotakts, wenn gilt HRS = 1 (Bild 4-78), und die Hälfte der Frequenz, wenn gilt HSR = 0.

Beispiel: Die Frequenz des Videotakts ist 14,5 MHz und die Zeit für die aktive Zone beträgt 40 µs, dann werden mit HSR = 1 580 Pixel pro Zeile, und mit HSR = 0 290 Pixel pro Zeile ausgegeben. Da sich die Zeitdauer der aktiven Zone nach den programmierten Horizontalkonstanten HC0 bis HC3 richtet, läßt sich die Zeitdauer daraus berechnen:

$$\text{Dauer der aktiven horizontalen Zone} = \frac{(HC2\text{-}HC1) \bullet 16 \bullet 2^{PSA}}{f_{(VCH)}}$$

$$\text{Pixel-Rate} = f_{(VCK)} \bullet 2^{(HSR-1)}$$

$$\text{Zahl der Pixel pro Zeile} = (HC2\text{-}HC1) \bullet 8 \bullet 2^{PSA} \bullet 2^{HSR} = (HC2\text{-}HC1) \bullet 2^{(PSA+HSR+3)}$$

Mit PSA = 0; HSR = 0 und HC2-HC1 = 40 erhält man den Wert 320 Pixel pro Zeile

4.7.3.5.2 Bildschirmgrenzen

Die horizontale Position von Objekten in der Bildschirmzeile wird durch die x-Koordinate festgelegt. Sie ist eine vorzeichenbehaftete 10-Bit-Zahl im Zweier-Komplementformat, die Werte zwischen -512 und +511 zuläßt. Dem linken Rand des Bildschirms wird die Position x = 0 zugewiesen. Im hochauflösenden Modus bedeutet der Zuwachs der x-Koordinate um Eins die Anzeige eines neuen Pixels. Wenn der hochauflösende Modus nicht gewählt ist (HSR = 1), bedeutet der Zuwachs der x-Koordinate die Anzeige von zwei Pixel nebeneinander. Die Position des rechten Bildschirmrandes ist daher:

$$x = (HC2\text{-}HC1) \bullet 2^{(PSA+3)}$$

Jeder Zeilenspeicher hat daher eine Pixel-Kapazität von x = 0 bis x = 320 (HSR = 1). Pixel, deren Koordinaten negative Werte aufweisen, werden nicht in den Zeilenspeicher aufgenommen, ebenso Pixel, deren Koordinaten den rechten Rand überschreiten. Der VSDD vergeudet keine Zeit, die Pixel-Koordinaten zu berechnen, die aus dem Rahmen fallen. Zur Vereinfachung errechnet der VSDD diesen Wert nicht. Das sollte die CPU übernehmen und die oberen sieben Bits des errechneten Werts in das Feld für die Bildschirmgrenzen im Bildschirmspeicher schreiben. Näheres dazu findet sich im Absatz über die Datenstrukturen. Es müssen nur die oberen sieben Bits des 10-Bit-Werts in den Speicher des VSDD geschrieben werden.

4.7.3.5.3 Transparente Pixel

Der Formatierer überprüft jedes Pixel auf Transparenz, bevor er es in die Zeile übernimmt. Transparente Pixel werden nicht in die Zeile aufgenommen. Die Methode, mit der man Pixel als transparent kennzeichnen kann, hängt davon ab, ob es sich um Charakter- oder Pixel-Objekte handelt.

Für Pixel-Objekte wird die Transparenz durch eine Eins im Bit TDE (Transparency Detect Enable) und durch Nullen in den Bits, die das Pixel beschreiben, gekennzeichnet. Ist das Bit TDE gelöscht, werden alle Bits auch die Nullen in den Zeilenspeicher geschrieben. Denn dem Kode, der aus lauter Nullen besteht, kann durchaus in der Farbtabelle eine Farbe zugeordnet werden, die daraufhin auch zur Anzeige kommt.

Für Charakter-Objekte gibt es verschiedene Wege, deren Pixel als transparent zu definieren. Das hängt vor allem davon ab, wie die Charakter im Bildschirmspeicher abgelegt sind. Eine genauere Beschreibung findet sich in den Datenstrukturen.

4.7.3.5.4 Die Farbtabelle

Die Farbtabelle im Innern des VSDD enthält 16 Farbmixturen, von denen jede aus zwölf Bits besteht und wovon wiederum vier Bits für die RGB-Farbe verwendet werden. Diese Farbmixturen werden durch die Ausgänge des Zeilenspeichers, der gerade die Anzeige steuert, adressiert. Somit sind es nicht die Ausgänge der Zeilenspeicher, die die Anzeige direkt steuern, sondern immer die Ausgänge der Farbtabelle. Es wurde bereits erwähnt, daß ein Pixel vier oder acht Bits umfassen kann. Werden acht Bits pro Pixel verwendet, werden nur die vier unteren Bits zur Adressierung der Farbtabelle verwendet. Ein Pixel-Kode von 0000_b wählt den ersten Eintrag und der Kode 1111_b den letzten Eintrag in der Farbtabelle. Dem dritten Kode, der durch 0010_b adressiert wird, kommt besondere Bedeutung zu: er wählt die Hintergrundfarbe. Daher ist der erste Schritt vor Beginn einer Zeilenkonstruktion, den gesamten Zeilenspeicher mit dem Wert 0010_b zu füllen, so daß nicht spezifizierte oder transparente Pixel immer in der Hintergrundfarbe erscheinen.

Diese intern vorhandene Farbtabelle wird nach jedem kompletten Bildaufbau neu mit Werten aus dem Bildschirmspeicher beschrieben. Somit sind die Farben durch Neubeschreiben der externen Eintragungen jederzeit änderbar.

4.7.3.5.5 Die Digital-Analog-Konverter (DAC)

Die Digital-Analog-Konverter erhalten ihre Daten aus der Farbtabelle und wandeln sie in analoge Signale um. Es gibt vier Digital-Analog-Konverter: einen für jede der drei Grundfarben und einen für das Einblendungssignal (OVR; Bild 4-80). Sie bestehen im wesentlichen aus einer Anordnung serieller Widerstände, verbunden mit Masse und einer Bezugsspannung (V_{ref}), und einem 4-zu-16-Dekoder, dessen Ausgänge über Transistoren eine dem Eingangskode entsprechende Analog-Spannung freischalten (Bild 4-81).

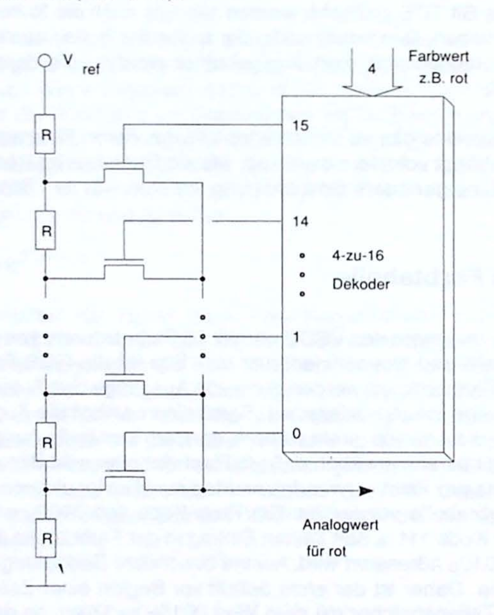


Bild 4-81. Ein Digital-Analog-Konverter

Die Stromtreibereigenschaft des aus einer Spannungsteilerschaltung entstammenden Signals ist nicht sehr hoch, d.h. jede Belastung des Analog-Signals würde dessen Wert verändern, womit es nicht mehr zu gebrauchen wäre. Daher ist eine externe Verstärkerschaltung notwendig, um den niederohmigen Eingang eines Farbmonitors anzusteuern. Ein Schaltungsvorschlag findet sich in Bild 4-82.

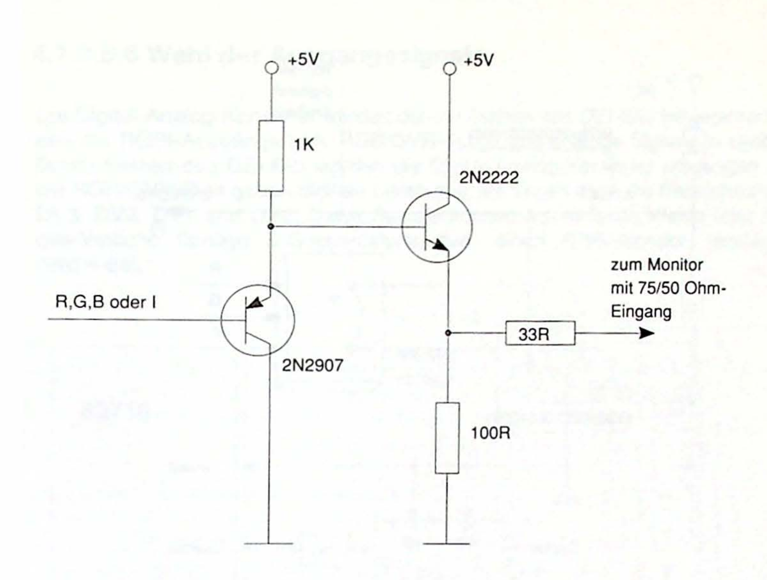


Bild 4-82. Verstärkerschaltung für die RGBI-Ausgänge

Für jede Farbe Rot, Grün und Blau ist ein eigener Digital-Analog-Konverter erforderlich. Wie bereits erwähnt, befindet sich in der Farbtabelle für jedes Pixel ein Bereich von zwölf Bits. Dieser Bereich speichert mit vier Bits die Information für den Rotanteil, in weiteren vier Bits die Information für den Grünanteil und mit den restlichen vier Bits die Information für den Blauanteil. Diese 4-Bit-Gruppen werden jeweils getrennt an die Eingänge der Digital-Analog-Konverter für die betreffende Farbe geführt.

Der vierte Digital-Analog-Konverter wird ebenfalls von den Ausgängen der Farbtabelle gesteuert. Immer, wenn die Farbtabelle das Bitmuster 0001 0001 0001_b ausgibt, zeigt der OVR-Pin mit High-Pegel die Farbe Weiß an. Jede andere Farbe bewirkt an diesem Ausgang Low-Pegel (Bild 4-83).

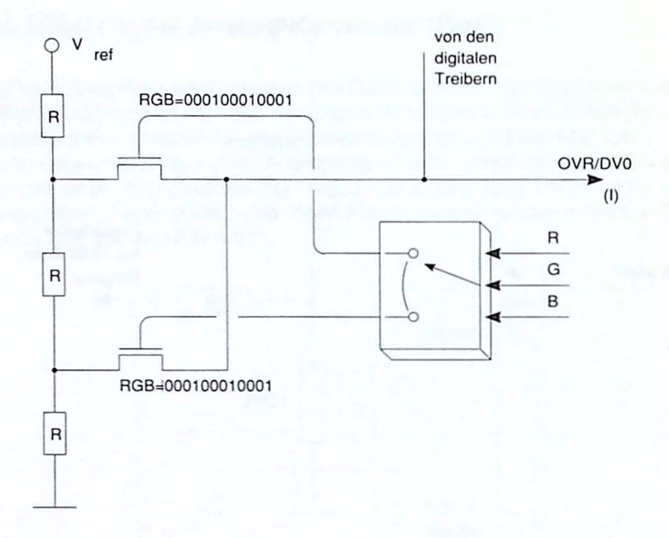


Bild 4-83. Der Digital-Analog-Konverter zur Einblendung in externe Signale

Das Signal an diesem Ausgang kann benutzt werden, um die Eingänge des Monitors von einer RGB-Quelle auf eine andere umzuschalten. Damit ist es möglich, ein vom VSDD erzeugtes Signal einem externen Signal zu überlagern. In der Praxis wird man zu diesem Zweck für den Hintergrund die Farbe Weiß wählen. Es muß also an der Adresse 0010_b der Farbtabelle der Wert 0001 0001 0001_b stehen. Gibt nun der VSDD keinen Bildpunkt an den Monitor, ist das extern erzeugte Bild unverändert zu sehen; füllt der VSDD den ganzen Bildschirm mit Pixel aus, ist das extern erzeugte Bild unsichtbar. Da diese Sonderfälle selten eintreten, wird sich im allgemeinen das vom VSDD erzeugte Bild vor das externe Bild schieben.

4.7.3.5.6 Wahl der Ausgangssignale

Die Digital-Analog-Konverter werden durch Löschen des DEI-Bits freigegeben, so daß die RGBI-Ausgänge (d.h. RGB/OVR-Ausgänge) analoge Signale ausgeben. Durch Setzen des DEI-Bits werden die Digital-Analog-Konverter umgangen und die RGB/OVR-Pins geben digitale Daten aus; sie tragen dann die Bezeichnungen DV3, DV2, DV1 und DV0. Diese Signale können auf einfache Weise über eine gewöhnliche 9polige D-Subminiaturbuchse einen RGBI-Monitor ansteuern (Bild 4-84).

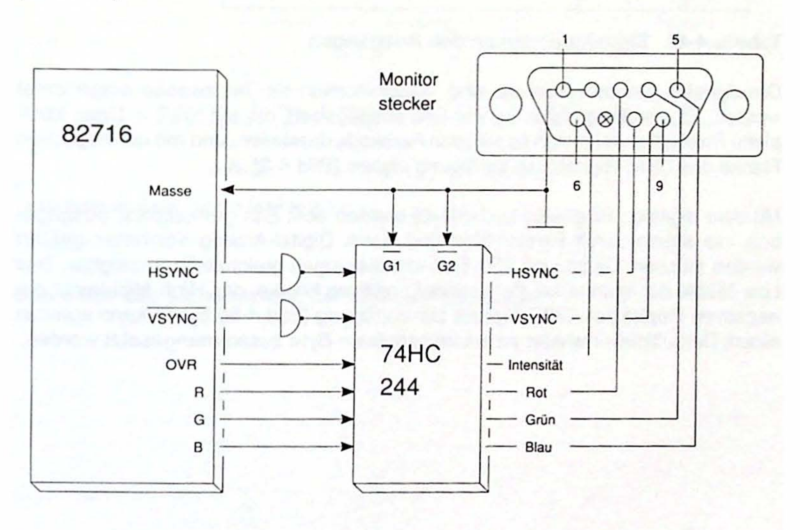


Bild 4-84. Der Anschluß der RGBI-Signale an den 9poligen Monitorstecker

Da gibt es noch zwei andere Bits, TF2 und TF1, die über die Art der ausgegebenen digitalen Daten entscheiden. Sind beide Bits gesetzt, wird der Pixel-Kode aus dem Zeilenspeicher unverändert an die Ausgänge geführt. Die anderen drei Kombinationen haben die Wirkung, daß einer der drei Farbkodes aus der Farbtabelle an die Ausgänge geführt wird. In den drei letzten Fällen wird eine monochrome "Anzeige" entstehen.

Die Wahlmöglichkeiten für die Ausgänge sind in Tabelle 4-41 zusammengefaßt.

DEI	TF2	TF1	Ausgänge	Signale
0	X	X	analog	RGBI
1	0	0	digital	rot monochrom
1	0	1	digital	grün monochrom
1	1	0	digital	blau monochrom
1	1	1	digital	Pixel-Kode

Tabelle 4-41. Signaloptionen an den Ausgängen

Die digitalen Monochrom-Modi sind hauptsächlich für Testzwecke eingerichtet worden. Es werden gleichzeitig vier Bits ausgegeben, die mit HSR = 1 das komplette Pixel oder, wenn man so will, den Farbkode darstellen, und mit der negativen Flanke des CKIO-Signals zur Verfügung stehen (Bild 4-85.a).

Mit dem digitalen Pixel-Modus (HSR=0) werden acht Bits gemultiplext ausgegeben, die extern durch Farbtabellen und durch Digital-Analog-Konverter geführt werden müssen. Damit sind 256 Farbschattierungen gleichzeitig anzeigbar. Das Low-Nibble der Information steht mit der positiven Flanke, das High-Nibble mit der negativen Flanke des CKIO-Signals zur Verfügung (Bild 4-85.b). Es kann somit in einem Demultiplexer wieder zu einem parallelen Byte zusammengesetzt werden.

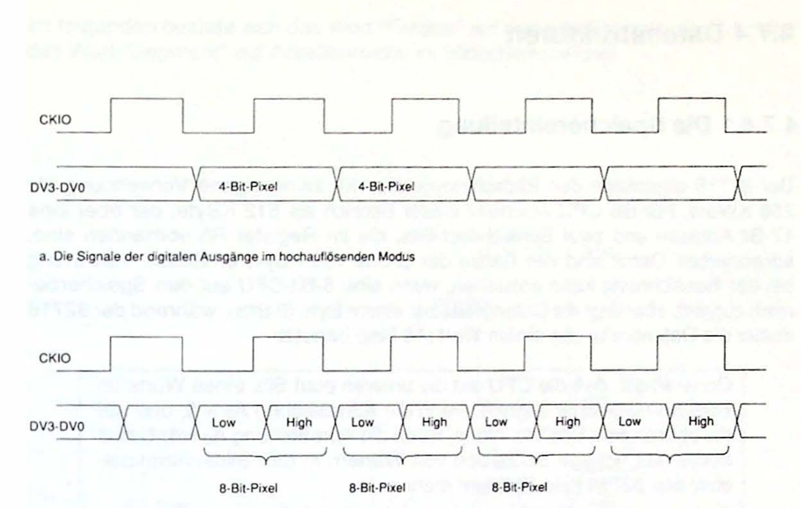


Bild 4-85. Die Signalformen der digitalen Ausgänge

4.7.4 Datenstrukturen

4.7.4.1 Die Speichereinteilung

Der 82716 organisiert den Bildschirmspeicher für seine eigene Verwendung als 256 KWord. Für die CPU erscheint dieser Bereich als 512 KByte, der über eine 17-Bit-Adresse und zwei Bank-Select-Bits, die im Register R5 vorhanden sind, adressierbar. Damit sind vier Bänke der Größe 128 KByte erhältlich. Verwirrung bei der Bezeichnung kann entstehen, wenn eine 8-Bit-CPU auf den Speicherbereich zugreift. Hier liegt die Datenbreite bei einem Byte (8 Bits), während der 82716 immer die Datenbreite von einem Wort (16 Bits) benutzt.

> Generell gilt, daß die CPU auf die unteren acht Bits eines Worts im Bildschirmspeicher zugreift, wenn die Adreßleitung $A_0 = 0$, und auf die oberen acht Bits des Worts, wenn die Adreßleitung $A_0 = 1$. Damit dürfte das richtige Schreiben von Wörtern in den Bildschirmspeicher des 82716 kein Problem mehr sein.

Der 82716 stellt der CPU für den Zugriff auf den Bildschirmspeicher zwei Fenster zur Verfügung. Durch das eine Fenster kann die CPU die Register, das sind Daten, die der 82716 für die eigene Steuerung benötigt, beschreiben (und lesen); es wird Registerfenster genannt. Durch das andere Fenster schreibt die CPU die eigentlichen Anzeigedaten, Charakter-Information, Objekt-Beschreibung etc.; es wird Datenfenster genannt.

Wie in Bild 4-86 gezeigt (vgl. Bild 4-62), steht das Datenfenster mit einem Bereich im Datensegment und das Registerfenster mit der festgelegten Stelle, dem Registersegment in Verbindung. Die Adressen, über die die CPU auf die gewünschten Stellen zugreift, sind verschiebbar und werden durch die CPU programmiert. Diese programmierten Daten sind Inhalt des Registersegments, das unveränderbar immer die untersten 16 Wörter (32 Bytes) des Bildschirmspeichers belegt.

Im folgenden bezieht sich das Wort "Fenster" auf den Adreßbereich der CPU und das Wort "Segment" auf Adreßbereiche im Bildschirmspeicher.

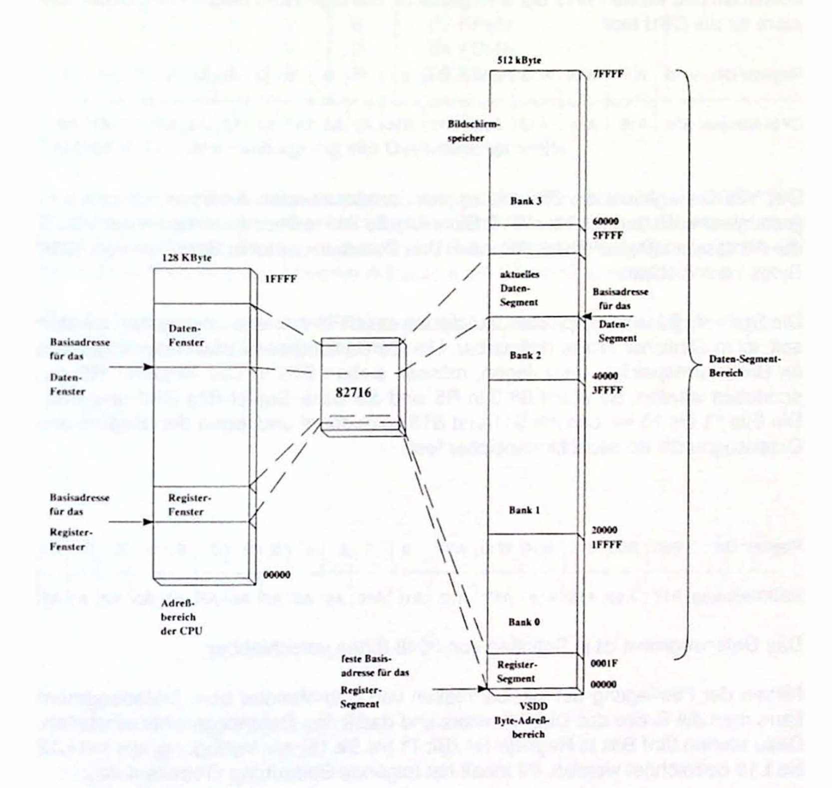


Bild 4-86. Der Zugriff auf den Bildschirmspeicher

4.7.4.2 Das Datenfenster

Die Basisadresse des Datenfensters wird in den Bits 11 - 15 des Registers R3 geschrieben und werden W12 bis W16 genannt. Sie legen den Beginn des Datenfensters für die CPU fest:

Register-Bits:	W16	W15	W14	W13	W12	0	0	0	0	0	0	0	0	0	0	0	0
CPU-Adresse:	A16	A15	A14	A13	A12	A11	A10	A9	A8	A7	A6	A5	A4	A3	A2	A1	A0

Der VSDD vergleicht die Bits $A_{12\text{-}16}$ jeder ankommenden Adresse mit den programmierten Werten W12 bis W16. Stimmen die Werte überein, erkennt der VSDD die Adresse als Datenfenster-Adresse. Das Datenfenster ist in Schritten von 4096 Bytes verschiebbar.

Die Stelle im Bildschirmspeicher, auf die mit der CPU-Adresse zugegriffen werden soll, ist in ähnlicher Weise definierbar. Um die Basisadresse des Datensegments im Bildschirmspeicher festzulegen, müssen sieben Bits in das Register R5 geschrieben werden. Bit 7 und Bit 8 in R5 sind die Bank-Select-Bits BS0 und BS1. Die Bits 11 bis 15 werden mit S11 und S15 bezeichnet und legen den Beginn des Datensegments im Bildschirmspeicher fest:

Register-Bits:	BS1	BS0	S15	W14	W13	W12	0	0	0	0	0	0	0	0	0	0	0	0
VSDD-Adresse:	A17	A16	A15	A14	A13	A12	A11	A10	A9	A8	A7	A6	A5	A4	A3	A2	A1	A0

Das Datensegment ist in Schritten von 2048 Bytes verschiebbar.

Neben der Festlegung der Basisadressen von Datenfenster bzw. Datensegment kann man die Breite des Datenfensters und damit des Datensegments einstellen. Dazu stehen fünf Bits in Register R4 (Bit 11 bis Bit 15) zur Verfügung, die mit L12 bis L16 bezeichnet werden. Ihr Inhalt hat folgende Bedeutung (Tabelle 4-42):

L16	L15	L14	L13	L12	Breite des Datenfensters
1	1	1	1	1	4 KByte
1	1	1	1	0	8 KByte
1	1	1	0	0	16 KByte
1	1	0	0	0	32 KByte
1	0	0	0	0	64 KByte
0	0	0	0	0	128 KByte

Tabelle 4-42. Die Festlegung der Datenfensterbreite

Um die absolute Bildschirmspeicheradresse zu ermitteln, werden die von der CPU eingehenden Adreßbits mit der programmierte Basisfensteradresse verglichen und geprüft, ob die Adresse in der mit den Bits L12 bis L16 eingestellten Breite ist. Ist das der Fall, wird die effektive Adresse nach Bild 4-87 ermittelt.

Man beachte, daß nach einem Reset noch kein Datenfenster existiert, es muß bei der Initialisierung festgelegt werden.

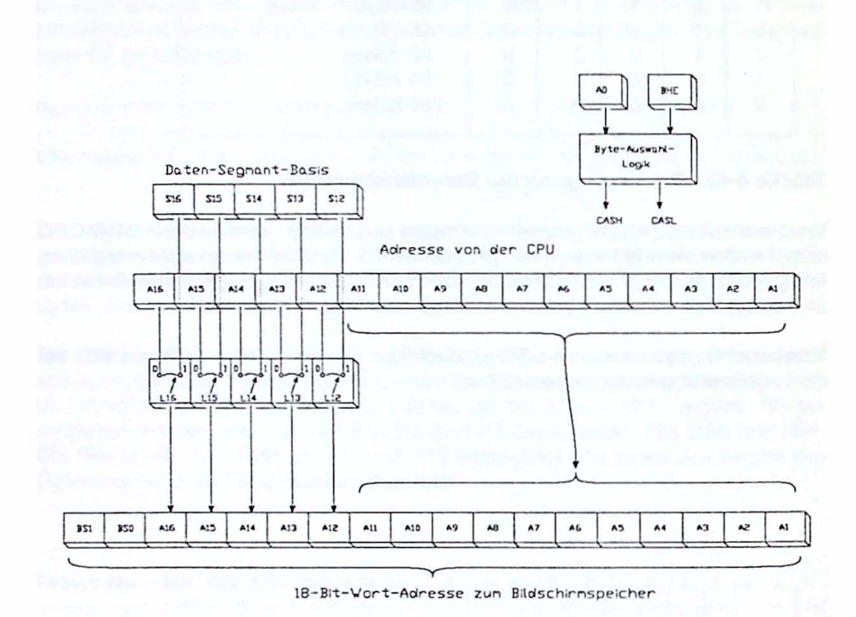


Bild 4-87. Die Übersetzung der CPU-Adresse in die Bildschirmadresse

4.7.4.3 Das Registerfenster

Das Registersegment ist ein Bereich im Bildschirmspeicher mit einer festen Länge und einer festen Adresse. Es umfaßt immer 16 Wörter und ist an den Wortadressen 00_h bis $0F_h$ angesiedelt. Für die CPU erscheinen die Register an den Byte-Adressen 00_h bis $1F_h$. Will die CPU zum Beispiel Register R15 beschreiben, muß sie das Low-Byte an Adresse $1E_h$ und das High-Byte an Adresse $1F_h$ richten. Die Adressen geben die Stellen im Bildschirmspeicher an. In ähnlicher Weise wie beim Schreiben von Daten existiert auch für die Register ein Fenster, dessen Anfangswert festgelegt werden kann. Anders als beim Datenfenster muß nach einem Reset dem Registerfenster ein fester Anfangswert zugewiesen sein, sonst wäre ein Beschreiben oder ein Umdefinieren der Basisadressen nicht möglich. Die Initialisierungsadresse hat den Wert 0400_h. Um diesen Wert den eigenen Gegebenheiten anzupassen, besteht die Möglichkeit, durch Beschreiben der Bits 4 bis 15 des Registers R2 die Basisadresse für das Registerfenster zu ändern:

Register-Bits:	R16	R15	R14	R13	R12	R11	R10	R9	R8	R7	R6	R5	0	0	0	0	0
CPU-Adresse:	A16	A15	A14	A13	A12	A11	A10	A9	A8	A7	A6	A5	A4	A3	A2	A1	A0

Alle Adressen von der CPU, die über das gleiche Bitmuster verfügen wie die Bits R5 bis R16, werden an das Registersegment Adresse 00 bis $1F_h$ weitergeleitet. Mit den verbleibenden Bits A0 bis A4 ist ein Bereich von 32 Bytes adressierbar, gerade soviel, wie den Registern Platz zur Verfügung steht.

Wenn der Bereich des Registerfensters den des Datenfensters überlagert, genießt das Registerfenster den Vorrang und die Daten werden in das Registersegment geschrieben.

4.7.4.4 Das Registersegment

Die wohl größte Bedeutung für die Steuerung des VSDD haben die 16 Register, aus denen das Registersegment besteht. Intern besitzt der 82716 nur Arbeitsregister, die nicht durch die CPU beschreibbar sind. Alle Instruktionen, die der 82716 für seine Arbeit benötigt, holt er sich aus diesem Bereich, der Bestandteil des Bildschirmspeichers ist und in seiner Lage unveränderbar an den Byte-Adressen von 00 bis $1F_h$ (00 bis $0F_h$ für Wortadressen) angesiedelt ist. Eine Übersicht über die Verwendung der 16 Register zeigt Tabelle 4-43.

Register	Inhalt		Byte- / Wortadresse	
R15	Horizontalkonstante 3	Vertikalkonstante 3	1E	0F
R14	Horizontalkonstante 2	Vertikalkonstante 2	1C	0E
R13	Horizontalkonstante 1	Vertikalkonstante 1	1A	0D
R12	Horizontalkonstante 0	Vertikalkonstante 0	18	0C
R11	Adreßzähler für die Zugriffstabelle		16	0B
R10	Basisadressen der Charakter-Generatoren		14	0A
R9	Basisadresse der Farbtabelle		12	09
R8	Basisadresse der Zugriffstabelle		10	08
R7	Basisadresse der Objekt-Beschreibungstabelle		0E	07
R6	Zahl der CPU-Zugriffe beim Zeilenaufbau		0C	06
R5	Basisadresse des Datensegments		0A	05
R4	Breite des Datenfensters		08	04
R3	Basisadresse des Datenfensters		06	03
R2	Basisadresse des Registerfensters		04	02
R1	Video-Konfigurationsregister 1		02	01
R0	Video-Konfigurationsregister 0		00	00

Tabelle 4-43. Die Inhalte der Register R0 bis R15

Es folgt eine Beschreibung der Inhalte der Register:

R0: Video-Konfigurationsregister 0

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DC2	DC1	DC0	BR4	BR3	BR2	BR1	BR0	DS1	DS0	DOF	HRS	DEN	SAB	DEI	UCF

UCF: (Update Controll Flag). Wenn es gesetzt ist, werden alle internen Arbeitsregister des VSDD nach jedem Bildschirmaufbau aus den entsprechenden Registern aktualisiert. Ist es gelöscht, liest der VSDD nur die Register R0 und R8. Bei der Initialisierung des Systems sollte mit dem ersten Speicherzugriff eine 0 in dieses Bit geschrieben werden, damit die restlichen Register korrekt beschrieben werden können, bevor der VSDD mit der eigenen Initialisierung beginnt.

DEI: (Digitally Encoded Information). Eine Eins bewirkt, daß die Ausgänge RGB und OVR digital sind, eine Null bewirkt die analoge Ausgabe.

SAB: (Slow Access Bit). Mit einer Eins wird die Zusammenarbeit mit langsamen DRAM-Bausteinen (t = 210 ns), mit einer Null die Zusammenarbeit mit schnellen DRAM-Bausteinen (t = 140 ns) ermöglicht.

DEN: (Display Enable). Eine Eins gestattet die Ausgabe der konstruierten Zeile an den Bildschirm, eine Null unterbindet sie. Die Synchronisationssignale werden nicht unterbunden, so daß der Bildschirm abgeschaltet wird. Das kann bei der Initialisierung oder beim Transfer größerer Datenmengen in den Bildschirmspeicher von Vorteil sein.

HRS: (High Resolution Screen). Gesetzt (1) wird die maximale horizontale Auflösung von 640 Pixel gewählt, gelöscht (0) bewirkt es die Ausgabe von 320 Pixel.

DOF, DS0, DS1: Mit diesen Bits wird dem VSDD die Größe des Speichers und die Organisation der Bausteine mitgeteilt. Über ihre Funktion gibt Tabelle 4-38 Auskunft.

BR0-BR4: (Blink Rate). Damit können ausgewählte Objekte zum Blinken veranlaßt werden (z.B. ein Cursor). Die Blinkrate ist ein Vielfaches von acht Bildschirmaufbauten. Der Multiplikator wird durch die Bits BR0 bis BR4 gegeben. Der Wert 00000 bewirkt das schnellste, 11111 das langsamste Blinken. Ist b der Wert in diesen fünf Bits und die Bildfrequenz 50 Hz, beträgt die Blinkrate 50:[8•(b+1)] Hz. Die Frequenzen können von 6 ¼ Hz bis 0,2 Hz variiert werden.

DC0-DC2: (Duty Cycle). Diese drei Bits bestimmen das Tastverhältnis der Blinkfrequenz. Es gelten die Werte nach Tabelle 4-44.

DC2	DC1	DC0	Auszeit	Einzeit
1	1	1	0%	100%
1	1	0	12,5%	87,5%
1	0	1	25,0%	75,0%
1	0	0	37,5%	62,5%
0	1	1	50,0%	50,0%
0	1	0	62,5%	37,5%
0	0	1	75,0%	25,0%
0	0	0	87,5%	12,5%

Tabelle 4-44 Blinkzeiten

R1: Video-Konfigurationsregister 1

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CH3	CH2	CH1	CH0	INL	MAS	SM	TMM	TMS	VP	EVC	PCE	FAE	RE	PSA	PRE

PRE: (Pipeline Read Enable). Eine 1 versetzt den VSDD in den Pipeline-Modus. er ist gedacht für Prozessoren, die über keinen eigen Ready-Eingang verfügen und somit keine Wait-States einlegen können. Eine 0 schaltet den Pipeline-Modus aus.

PSA: (Prescaler Active). Diese Bit bestimmt die Relation zwischen der Frequenz des Videotakts und der des Synchronisierungsgenerators. Die Wirkung des Bits zeigt Bild 4-65 und Bild 4-78.

RE: (Read Enable). Eine 1 erlaubt das Lesen von Daten aus dem Bildschirmspeicher durch die CPU, eine Null verhindert es. Das Schreiben von Daten wird nicht beeinflußt und ist immer möglich. Einen Sinn hat diese Option, wenn der Adreßbereich, den der VSDD beansprucht, mit dem Adreßbereich des CPU-Programmspeichers kollidiert. In einem solchen Fall wird das RE-Bit gelöscht und die CPU erhält die Daten aus dem Programm- und nicht aus dem Bildschirmspeicher. Bei der Adressierung durch Mikrocontroller entfällt dieser Umstand, da sie über getrennte Steuerleitungen für Lesezugriffe auf das ROM und das RAM besitzen.

FAE: (Free Access Enable). Mit einer 1 wird der RDY-Pin des VSDD umdefiniert. Er zeigt an, daß der 82716 frei für einen Prozessorzugriff ist. Mit einer 0 gibt der RDY-Pin das Ready-Signal an die CPU aus.

PCE: (Priority Counter Enable). Um die Zeilenkonstruktion durch Prozessorzugriffe auf den Bildschirmspeicher nicht unnötig zu verzögern, kann dieses Bit gesetzt werden. Es wird dann während der Zeilenkonstruktion nur die Zahl an Zugriffen erlaubt, die in Register R6 genannt ist. Wird diese Zahl überschritten, stoppt der 82716 über die Ready-Leitung die Aktivitäten der CPU. Dazu muß das Bit FAE gelöscht sein und die CPU über einen Ready-Eingang verfügen.

EVC: (External Video Clock). Wenn es gesetzt ist, wird der Pin CKIO zum Eingang und akzeptiert externe Videosignale. Eine 0 schaltet auf den internen Oszillator um (Bild 4-65).

VP: (Vertical Pulse). Mit einer 0 entsprechen die VSYNC-Impulse dem europäischen TV-Standard, wenn der Interlaced-Modus gewählt ist. Am Ende eines Bildschirmaufbaus wird zur Länge des vertikalen Synchronisierungsimpuls die Hälfte einer Zeilenzeit addiert. Das benötigt der Monitor für Synchronisierungszwecke.

TMS,TMM: (Twin Mode Master/Slave). Damit können zwei Bausteine in den Zwillings-Modus versetzt werden. Dabei erzeugen 82716 (mit verschiedenen Bildschirmspeichern) alternative Zeilen auf derselben Anzeige. Aus Synchronisationsgründen muß einer zum Master (TMM = 1 und TMS = 0), der andere zum Slave (TMM = 0 und TMS = 1) ernannt werden.

SM: (Sync Mode). Gesetzt läßt es den VSDD im Composite-Modus arbeiten. Dabei gibt der Pin HSYNC sowohl das vertikale als auch das horizontale Synchronisierungssignal aus.

MAS: (Master Sync). Wenn es gelöscht ist, akzeptiert der VSDD externe Synchronisierungssignale und klinkt sich über den internen PLL-Schaltkreis in sie ein. Wenn es gesetzt ist, sind die Pins HSYNC und VSYNC Ausgänge.

INL: (Interlacing). Mit einer 1 wird der VSDD in den Interlaced-Modus versetzt. Eine 0 läßt ihn im normalen Modus arbeiten.

CH0-CH3: (Char Height). Diese vier Bits stellen die Zahl an Bildschirmzeilen dar, die für ein Charakter-Zeichen aufgewendet werden sollen. Es sind Eintragungen von 0000 bis 1111_b möglich, wobei der Wert 0000 nicht die Höhe 0, sondern die Höhe 16 bedeutet. Wenn die Charakter-Höhe zu 10, 12, 14 oder 16 Zeilen programmiert ist, können auch Charakter, die doppelt hoch definiert sind (20, 24, 28 oder 32), ebenfalls angezeigt werden. Näheres in Absatz 4.7.4.7.2.

R2: Basisadresse des Registerfensters

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R16	R15	R14	R13	R12	R11	R10	R9	R8	R7	R6	R5	PLL	TF2	TF1	ME

ME: (Margin Enable). Wenn dieses Bit gesetzt ist und der hochauflösende Modus nicht gewählt ist, erzeugt der VSDD eine Randanzeige in der Hintergrundfarbe. Diese Möglichkeit ist für solche Anwendungen beabsichtigt, die Standard-TV-Sets benutzen und bei denen die Darstellungspräzision einer Bildschirmzeile beschränkt ist. Mit ME = 0 werden die RGB-Signale außerhalb der aktiven Zone abgeschaltet, mit ME = 1 erscheint der Rand in der Hintergrundfarbe.

TF1,TF2: (Test Flag). Wurde der digitale Ausgabemodus (DEI = 1) gewählt, bestimmen diese Bits nach Tabelle 4-41 die Art der ausgegebenen Daten.

PLL: (Phase Locked Loop). Ist diese Bit gelöscht, dann zeigt der Ausgang PLLCTL mit Low-Pegel an, daß das interne Synchronisierungssignal im Vergleich zum externen zu langsam ist, und mit High-Pegel, daß es zu schnell ist. Ist das Bit gesetzt, wird der Ausgang PLLCTL nur zwischen den negativen Flanken von internem und externen Synchronisierungssignal aktiv und zeigt mit High-Pegel ein zu langsames, mit Low-Pegel ein zu schnelles internes Signal an. Für den Rest der Zeit ist der Ausgang im Tri-State. Die Impulslänge ist also zur Phasenverschiebung proportional und kann zur Justierung eines LC-Oszillators benutzt werden.

R5-R16: Mit diesen Bits wird die Basisadresse der Registerfensters für die CPU festgelegt. Weist es beispielsweise den Inhalt 000110111001 auf, findet die CPU das Register R2 an den Adressen 3724_h (Low-Byte) und 3725_h (High-Byte) vor.

R3: Basisadresse des Datenfensters

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
W16	W15	W14	W13	W12	0	SB9	SB8	SB7	SB6	SB5	SB4	SB3	0	0	0

SB3-SB9: (Screen Boundary). Diese Bits stellen die oberen zehn Bits der x-Koordinate des rechten Bildschirmrandes dar. Die restlichen drei Bits eines Pixels werden bei der Zeilenkonstruktion hinzugerechnet. Überschreiten die oberen zehn Bits eines Pixels diesen voreingestellten Wert, werden sie nicht mehr angezeigt. Bei der Initialisierung des Bausteins ist hierin besonders sorgfältig zu verfahren.

W12-W16: Mit diesen Bits wird die Basisadresse des Datenfensters festgelegt.

R4: Breite des Datenfensters

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
L16	L15	L14	L13	L12	0	0	0	0	0	0	0	0	0	0	0

L12-L16: Mit diesen Bits wird die Breite nach Tabelle 4-42 des Datenfensters festgelegt und damit auch die Breite des Datensegments. Wenn die Fensterbreite auf 128 KByte eingestellt ist, ist der VSDD mit jeder Adresse ansprechbar. Das ist besonders bei Mikrocontrolleranwendungen von Vorteil, bei denen das Programm im Chip integriert ist.

R5: Basisadresse des Datensegments

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
S15	S14	S13	S12	S11	0	0	BS0	BS1	0	0	0	0	0	0	0

BS0,BS1: (Bank Select). Bit diesen beiden Bits wird der 512-KByte-Speicher in vier Bänke zu je 128 KByte aufgeteilt. In einer 256-KWord-Adresse, die 18 Bits benötigt stellt Bit BS0 die Adreßleitung A16 und BS1 die Adreßleitung A17 dar. Nur die Daten von Pixel-Objekten können in allen vier Bänken untergebracht sein. Alle anderen Anzeigedaten müssen sich in Bank 0 befinden.

S11-S15: Mit diesen Bits wird Basisadresse des Datensegments festgelegt.

R6: Zahl der CPU-Zugriffe beim Zeilenaufbau

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	PQ3	PQ2	PQ1	PQ0

PQ0-PQ3: Mit diesen Bits wird die Maximalzahl an Speicherzugriffen durch die CPU während eines Zeilenaufbaus festgelegt. Diese Einschränkung kann erst dann wirksam werden, wenn das Bit PCE in R1 gesetzt ist.

R7: Basisadresse der Objekt-Beschreibungstabelle

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
A15	A14	A13	A12	A11	A10	A9	A8	A7	A6	0	0	0	0	0	0

A6-A15: Diese Register beinhalten die Wort-Basisadresse der Objekt-Beschreibungstabelle im Bildschirmspeicher. Da zur Adressierung eines 256-KWord-Speichers 18 Adreßbits nötig sind und der VSDD die zwei höchsten Bits auf Null setzt, muß sich die Objekt-Beschreibungstabelle in Bank 0 befinden.

R8: Basisadresse der Zugriffstabelle

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
B15	B14	B13	B12	B11	B10	B9	B8	B7	B6	B5	B4	B3	B2	B1	B0

B0-B15: Diese Register beinhalten die Wort-Basisadresse der Zugriffstabelle im Bildschirmspeicher. Da zur Adressierung eines 256-KWord-Speichers 18 Adreßbits nötig sind und der VSDD die zwei höchsten Bits auf Null setzt, muß sich die Zugriffstabelle in Bank 0 befinden. Dieses Register wird nach jedem erfolgten Bildaufbau in den 82716 gelesen.

R9: Basisadresse der Farbtabelle

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
D15	D14	D13	D12	D11	D10	D9	D8	D7	D6	D5	D4	D3	D2	D1	D0

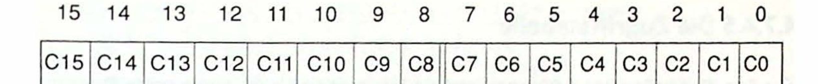
D0-D15: Diese Bits geben die Basisadresse der Farbtabelle an. das es sich um eine Wortadresse handelt und die beiden höchsten Adreßbits auf Null gesetzt werden, muß sich die Farbtabelle in Bank 0 befinden.

R10: Basisadresse der Charakter-Generatoren

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	G13	G12	G11	G10	G03	G02	G01	G00

G00-G13: G0 bzw. G1 gibt die Basisadresse des Charakter-Generators 0 bzw. des Generators 2 an. Dabei entsprechen die Bits Gx0 bis Gx3 den Adreßbits A12 bis A15. Somit kann der Anfang eines jeden Generators nur in Schritten von 2048 Bytes geändert werden. Auch bei dieser Adressierung werden die zwei höchsten Adreßbits mit Nullen gefüllt, so daß die Charakter-Generatoren in Bank 0 angesiedelt werden müssen.

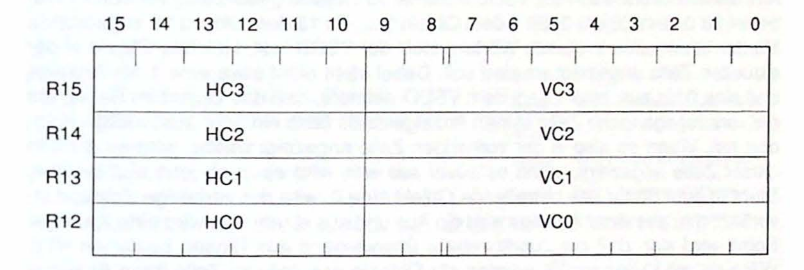
R11: Adreßzähler für die Zugriffstabelle



15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
C15	C14	C13	C12	C11	C10	C9	C8	C7	C6	C5	C4	C3	C2	C1	C0

C0-C15: Diese Bits weisen auf den nächsten Eintrag in der Zugriffstabelle. Da diese Register vom VSDD als temporäres Zwischenregister verwendet und von ihm aktualisiert wird, sollte es von er CPU nicht beschrieben werden.

R12-R15: Horizontal- und Vertikalkonstanten



	15–9	8–0
R15	HC3	VC3
R14	HC2	VC2
R13	HC1	VC1
R12	HC0	VC0

HCn,VCn: Diese Bits sind Bildschirmkonstanten, die vom Synchronisierungsgenerator verwendet werden. Über die Verwendung der Zahlen und deren Berechnung wird ausführlich in Absatz 4.7.3.4 geschrieben.

4.7.4.5 Die Zugriffstabelle

Die Zugriffstabelle (Bild 4-88) speichert die Information für die senkrechte Position eines jeden Objekts. Daher muß die Zahl an Wörtern in dieser Tabelle so groß wie die Zahl der aktiven Bildschirmzeilen sein. Die Basisadresse dieser Tabelle findet sich Register R8. Jedes Wort wird einer Bildschirmzeile zugeordnet. Das Wort, das an der Basisadresse der Tabelle steht, korrespondiert mit der obersten Bildschirmzeile, das folgende Wort mit der zweiten Zeile von oben usw.

Ein Wort besteht aus 16 Bits. Jedes Bit in diesem Wort repräsentiert ein Objekt. Aus diesem Grund kann der VSDD maximal 16 Objekte gleichzeitig verwalten. Dabei ist Bit 0 dem Objekt 0, Bit 1 dem Objekt 1,..., Bit 15 dem Objekt 15 zugeordnet. Mit der Information in diesen Wörtern stellt der VSDD fest, welches Objekt in der aktuellen Zeile angezeigt werden soll. Dabei steht nicht etwa eine 1 für Anzeige und eine 0 für aus; eine 1 sagt dem VSDD vielmehr, daß das Objekt im Bezug auf die vorausgegangene Zeile seinen Anzeigestatus (also ein oder aus) **nicht** verändert hat. Wenn es also in der vorherigen Zeile angezeigt wurde, wird es auch in dieser Zeile angezeigt; wenn es zuvor aus war, wird es auch jetzt aus bleiben. Steht in dem Bit für das betreffende Objekt eine 0, wird der vorherige Zustand invertiert, d.h. aus einer Anzeige wird ein Aus und aus einem Aus wird eine Anzeige. Somit wird klar, daß die Zugriffstabelle überwiegend aus Einsen bestehen wird. Wäre sie mit Nullen gefüllt, würden alle Objekte von Zeile zu Zeile ihren Anzeigestatus ändern. Am Ende eines kompletten Bildaufbaus werden automatisch alle Objekte ausgeschaltet, so daß ein darzustellendes Objekt für jeden Bildschirm ausdrücklich eingeschaltet werden muß.

Um ein Objekt einzuschalten, muß in dem Wort, das zur ersten anzuzeigenden Zeile des Objekts gehört, in das entsprechende Bit eine Null geschrieben werden. Es folgen nun solange Einsen, bis das Objekt wieder ausgeschaltet werden soll. Um es auszuschalten, muß in dem der letzten Anzeigezeile folgenden Wort wiederum eine Null stehen. Diese Technik gestattet es, ein Objekt mehrmals auf dem Bildschirm erscheinen zu lassen. Am Ende des gesamten Bildaufbaus werden alle Objekte, die noch aktiv sind, ausgeschaltet, so daß sie zu Beginn des neuen Bildschirms ausdrücklich aktiviert werden müssen.

Eine Bewegung von Objekten in senkrechter Richtung erreicht man durch Verschieben der beiden Nullen von Wort zu Wort in der Zugriffstabelle. Diese Bewegung wird Scrolling genannt.

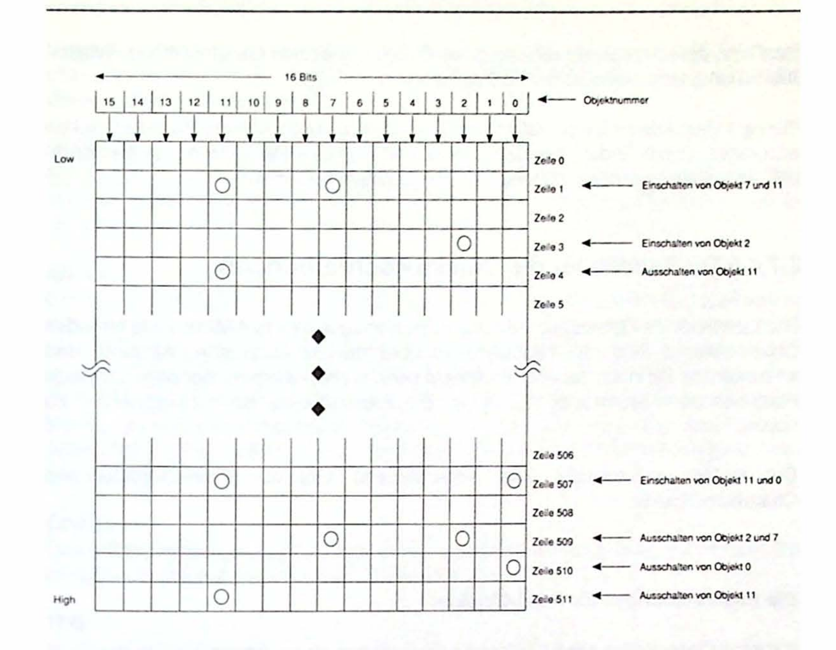


Bild 4-88. Die Struktur der Zugriffstabelle

Mit Hilfe des Registers R11 steuert der VSDD seine Zugriffe auf diese Tabelle. Es stellt nichts anderes als ein Adressen-Offset dar, das nach jeder Zeilenkonstruktion um Eins erhöht wird und vor jeder Neukonstruktion zur Basisadresse (in Register R8) addiert wird. Es wird nach einem kompletten Bildaufbau gelöscht. Das so adressierte Wort der Zugriffstabelle wird in den VSDD gelesen und Bit für Bit auf Änderungen, also auf Nullen geprüft.

Soll das Objekt n angezeigt werden, wird das Beschreibungsfeld des Objekts gelesen. Die Adresse, in der die Objektbeschreibung zu finden ist, erhält der VSDD auf folgende Weise: Er nimmt die Bits aus Register R7, fügt den Binärkode n3-n0, der die Nummer des Objekts enthält, hinzu und setzt am linken Ende zwei, am rechten Ende drei Nullen an:

0	0	A15	A14	A13	A12	A11	A10	A9	A8	A7	A6	n3	n2	n1	n0	0	0	0

Adresse für die Objekt-Beschreibungstabelle

Der Platz, der für die Beschreibung eines Objekts reserviert ist, umfaßt vier Wörter. Alle vier Wörter werden vom VSDD gelesen.

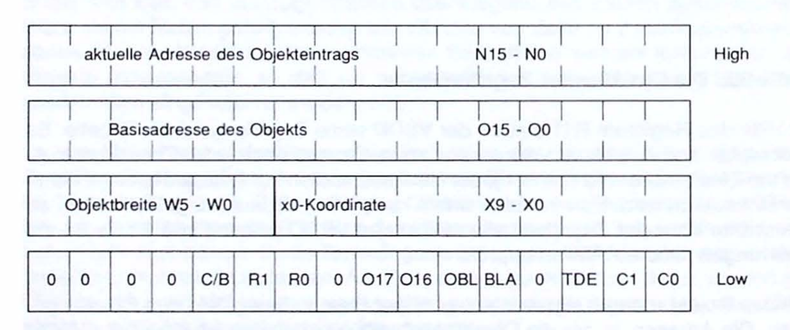
Für sehr viele Anwendungen ist es empfehlenswert, verschiedene Zugriffstabellen anzulegen. Durch Ändern der Basisadresse in Register R8 kann dabei auf einfache Art auf einen anderen "Bildschirm" umgeschaltet werden.

4.7.4.6 Die Tabelle für die Objektbeschreibungen

Die Tabelle für die Objektbeschreibungen besteht aus einem 4-Wort-Feld für jedes Objekt. Dieses Feld hält Informationen über die Basisadresse, Attribute und x-Koordinate. Da in der Tabelle 16 Objekte beschrieben werden, benötigt sie einen Platz von 64 Wörtern oder 128 Bytes. Die Basisadresse ist in Register R7 zu finden.

Der 82716 unterscheidet zwei verschiedene Objekte: Pixel-Objekte und Charakter-Objekte.

Die Beschreibungen für Pixel-Objekte:



N0-N15; O0-O15; O16,O17:
Die Basisadresse O0-O15 zeigt auf den Beginn des Datenbereichs für das betreffende Objekt. Hier stehen nur 16 Bits. Um die Daten in jeder Bank erreichen zu können, werden bei der Adreßberechnung die Bits O16 und O17 zur Bankauswahl hinzugefügt. Die Basisadresse O0-O15 wird zu Beginn eines Bildaufbaus in das darüberstehende Wort N0-N15 kopiert, das als Arbeitsregister für den VSDD dient.

Hieraus entnimmt er die aktuelle Adresse des Pixel-Eintrags im Bildschirmspeicher und beschreibt es mit dem nächsten Wert für die folgende Zeilenkonstruktion. Dieses Wort sollte von der CPU nicht geändert werden.

W0-W5:
Mit diesen sechs Bits wird die Größe eines Objekts in 4-Wort-Einheiten im Bildschirmspeicher angegeben. Die Zahl 000001 bedeutet eine Größe von 1 = 4 Wörter = 8 Bytes = 64 Bits; 111111 bedeutet die Größe von 252 Wörtern.

X0-X9:
Der Inhalt bildet eine vorzeichenbehaftete 10-Bit-Zahl, die die horizontale Position des linken Rands eines Objekts angibt. Der Wert kann zwischen -512 (=1111111111$_d$) und +511 (=0111111111$_d$) liegen. Das bedeutet, daß im niedrig auflösenden Modus mit 320 Pixel pro Zeile, Objekte in Ein-Pixel-Schritten zwischen den x-Koordinaten von -512 bis +512 bewegt werden können. Im hochauflösenden Modus ist diese Bewegung in Zwei-Pixel-Schritten zwischen den Koordinaten -1024 und +1022 möglich. Pixel, die rechts oder links des Bildrandes liegen, werden nicht angezeigt.

C0,C1:
Diese Bits stellen für Objekte, die mit nur zwei Bits pro Pixel gespeichert sind, die fehlende Farbinformation für die Farbtabelle dar.

TDE:
Ist dieser Inhalt eine Eins, werden Pixel, die den Wert 0000 aufweisen, nicht in den Zeilenspeicher geschrieben. Der Speicher behält seine ursprüngliche Information, so daß Objekte niedrigerer Priorität hinter den Objekten höherer Priorität zu sehen sind. Ist der Inhalt ds Bits eine Null, wird auch der Wert 0000 in den Speicher übernommen, der die alten Werte überschreibt. Dem Kode 0000 kann in diesem Fall eine Farbe zugeordnet werden.

BLA:
Der Inhalt entscheidet, ob das Objekt angezeigt werden soll. Ist der Wert eine Eins, ist das Objekt ausgeschaltet.

OBL:
Mit einer Eins in diesem Bit läßt der VSDD das Objekt mit den in Register R0 voreingestellten Werten blinken.

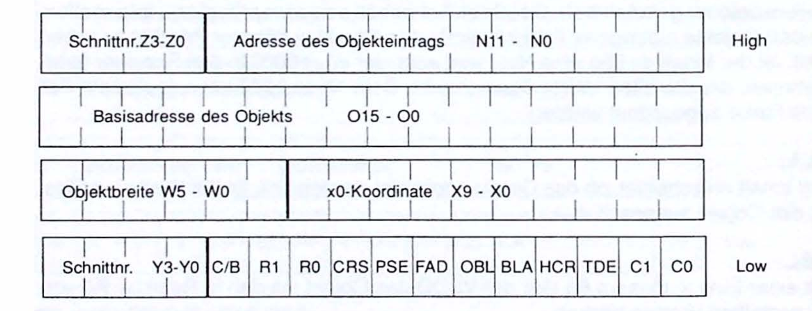
R0,R1:
Diese beiden Bits teilen dem VSDD mit, wieviele Bits pro Pixel im Bildschirmspeicher abgelegt sind (Tabelle 4-45). Die genaue Zahl hängt vom Wert des HRS-Bits in Register R0 ab.

HRS	R1	R0	Zahl der Bits pro Pixel
0	0	0	ungültig
0	0	1	2
0	1	0	4
0	1	1	8
1	0	0	ungültig
1	0	1	ungültig
1	1	0	2
1	1	1	8

Tabelle 4-45. Die Festlegung der Pixel-Zahl (C/B=0)

C/B:
Diese Bit entscheidet über die Art des Objekts: Eine Null definiert ein Pixel-Objekt, eine Eins ein Charakter-Objekt.

Die Beschreibungen für Charakter-Objekte:



N0-N11; O0-O15:
Für Charakter-Objekte wird mit den Bits N0 bis N11 die Startadresse der aktuellen Textzeile festgelegt. Wenn die letzte Zeile einer Textdarstellung beendet ist, wird dieser Inhalt so aktualisiert, daß er auf die nächste Textzeile im Objekt zeigt. Nach Beendigung des Bildaufbaus werden die Bits O0 bis O11 in die Bits N0 bis N11 kopiert. Deswegen kann ein Charakter-Objekt nicht größer als vier KWord sein.

W0-W5:
Mit diesen sechs Bits wird die Größe eines Objekts in 4-Wort-Einheiten im Bildschirmspeicher angegeben. Die Zahl 000001 bedeutet eine Größe von 1 = 4 Wörter = 8 Bytes = 64 Bits; 111111 bedeutet die Größe von 252 Wörtern. Diese 4-Wort-Einheit kann acht Charakter aufnehmen, wenn der Modus 1-Byte-pro-Charakter gewählt ist, und etwas mehr als zwei Charakter und Attribute, wenn der Modus 3-Byte-pro-Charakter gewählt ist.

X0-X9:
Diese Bits haben dieselbe Bedeutung wie bei der Pixel-Beschreibung.

Y0-Y3:
Ein Charakter-Zeichen weist eine gewisse Höhe auf, die im VSDD durch das Register R1 festgelegt wird. Zur Darstellung eines Buchstabens werden somit mehrere Bildschirmzeilen benötigt und jede Bildschirmzeile bildet einen Schnitt durch das Charakter-Zeichen. Diese Schnitte sind numeriert. Die Zahl 0 bedeutet den Schnitt durch den Fußpunkt des Zeichens. Die Nummer des obersten Schnitts ist die Zahl, die in Register R0 als Charakter-Höhe festgelegt ist. Damit also ein Zeichen in der üblichen Form ausgegeben wird, muß der Wert in den Y-Bits identisch mit dem Wert in R0 sein. Jeder andere Wert in diesen Bits läßt das Charakter-Zeichen zwischen Kopf- und Fußpunkt beginnen. Durch Veränderung des Inhalts kann eine Textzeile gescrollt werden.

Z0-Z3:
Die Bits Z0 bis Z3 stellen den Arbeitsbereich für den VSDD bei der Errechnung der nächsten Schnittnummer eines Zeichens dar. Zu Beginn eines Textbereichs wird der Inhalt von Y0 bis Y3 in die Bits Z0 bis Z3 kopiert, wo sie nach jeder Bildschirmzeile um Eins vermindert werden. Tritt dabei ein Unterlauf ein, erfolgt ein Nachladen aus den Bits des Register R1.

C0,C1:
Für Charakter-Objekte, die im Modus 1-Byte-pro-Charakter abgelegt sind, werden die beiden Bits als die beiden höherwertigen Bits für die Farbauswahl hinzuaddiert. Dabei gilt folgendes:
Vordergrundfarbe: C1,C0 = 0 1
Hintergrundfarbe: C1,C0 = 0 0

TDE; BLA; OBL:
Diese Bits haben dieselbe Bedeutung wie bei den Pixel-Objekten.

FAD:
Wenn es gesetzt ist, heißt das, daß für die Charakter-Beschreibung 3 Bytes verwendet werden. Jeder Charakter besteht dann aus einem Byte für den ASCII-Kode und zwei weitere Bytes für Attribute. Wenn es gelöscht ist, heißt das, daß die Charakter-Beschreibung nur ein Byte umfaßt. Jeder Charakter wird als ASCII-Kode gespeichert.

HCR:
Dieses Bit hat nur dann eine Wirkung, wenn das FAD-Bit gesetzt ist. Weist das HCR-Bit eine Eins auf, wird die Farbe für das Zeichen und den Hintergrund aus 16 Farben gewählt. Ist das HCR-Bit gelöscht, stehen nur acht Farben zur Verfügung, das Zeichen wird aber in doppelter Höhe dargestellt.

CRS:
Wenn das FAD-Bit und das CRS-Bit gesetzt sind, wird dem MSK-Bit, das als Attribut dem ASCII-Kode beigefügt ist, erlaubt, das Charakter-Zeichen unsichtbar zu machen. Wenn das FAD-Bit gelöscht ist, kann mit dem CRS-Bit von einem zum anderen Charakter-Generator umgeschaltet werden.

PSE:
Eine Eins erlaubt die proportionale Darstellung von Zeichen.

C/B:
Für die Wahl der Charakter-Beschreibung muß das Bit gesetzt werden.

R0,R1:
Diese beiden Bits geben dem VSDD Information über die Breite der darzustellenden Zeichen (Tabelle 4-46). Die genaue Breite hängt vom Wert des HRS-Bits in Register R0 ab.

HRS	R1	R0	Pixel pro Charakter
0	0	0	6
0	0	1	8
0	1	0	12
0	1	1	16
1	0	0	16
1	0	1	6
1	1	0	8
1	1	1	12

Tabelle 4-46. Die Festlegung der Zeichenbreite (C/B=1)

4.7.4.7 Die Objektdaten

Objekte erscheinen rechtwinkelig und aufrecht auf dem Bildschirm. Die Objekt-Daten beginnen an der Adresse, die in der Objekt-Beschreibungstabelle genannt ist. Der Speicherbereich, der von den Objekt-Daten eingenommen wird, hängt von der Höhe und Breite des Objekts ab und von der Art, wie dicht die Daten gespeichert sind. Die minimale Länge beträgt vier Wörter und kann nur in diesen Schritten vergrößert werden.

Die Objekt-Breite wird durch die Bits W0 bis W5 bestimmt. Auch hier ist wiederum die Einheit vier Wörter. Ist die Eintragung beispielsweise 0010101_b, dann ist das Objekt 10 x 4 = 40 Wörter breit oder 80 Bytes. Im folgenden wird auf diese Objekt-Breite Bezug genommen.

Objekt-Typ	Min. Breite	Max. Breite
2 Bit pro Pixel	32 Pixel	2016 Pixel
4 Bit pro Pixel	16 Pixel	1008 Pixel
8 Bit pro Pixel	8 Pixel	504 Pixel
1 Byte pro Zeichen	8 Zeichen	504 Zeichen
3 Byte pro Zeichen	1 Zeichen	168 Zeichen

Tabelle 4-47. Minimale und maximale Objekt-Breiten

4.7.4.7.1 Pixel-Objekte

Mit den Bits R0 und R1 in der Pixel-Beschreibungstabelle wird die Zahl der Bits pro Pixel festgelegt. Jede Bitzahl, die so festgelegt ist, beansprucht einen Platz im DRAM. Für diese Bits wird je Pixel ein Wort zur Verfügung gestellt. Da mit den Bits N0 - N15 Wörter adressiert werden können, ist es nicht möglich, die recht lockere Struktur der Bits in den Wörtern zu komprimieren. Sind vier Bits pro Pixel definiert, werden nur vier Bits eines Speicherworts benutzt, die restlichen bleiben frei. Die benutzten Bits müssen an die unterste Stelle des Worts geschrieben werden. Entsprechend werden zwei oder acht Bits pro Pixel behandelt.

Wenn das Objekt eine Breite (W) von zehn 4-Wort-Einheiten besitzt und vier Bits pro Pixel definiert sind, dann beträgt die Breite des Objekts 10•16 = 160 Pixel. Diese 160 Pixel sind in 40 Wörtern abgelegt. Das erste Wort ist an der Objekt-Basisadresse (O), die in der Beschreibungstabelle festgelegt ist, zu finden; das letzte Wort an der Adresse, die sich aus (O + 4•W) - 1 errechnet.

Beispiel: O = 1234_h, die Bits O16 und O17 haben den Inhalt 01 und W = 10. Dann ist die Basisadresse des Objekts 11234_h und die letzte Pixel-Adresse $1125B_h$. Der VSDD liest also die Informationen für die erste Zeilenkonstruktion aus den Adressen 11234_h bis $1125B_h$. Anschließend werden in der Beschreibungstabelle die Bits N0 bis N15 aktualisiert, indem der VSDD den Wert $125C_h$ reinschreibt. Die Daten für die zweite Bildschirmzeile werden also aus den Adressen von $1125C_h$ bis 11283_h geholt und der VSDD beschreibt die Bits N0 bis N15 mit dem Startwert 1284_h für die dritte Zeile usw. Da die Bits O16 und O17 bei dieser Adreßberechnung unverändert bleiben, kann das Objekt die 64-KWord-Grenze nicht überschreiten.

Erste Zeile:	11234_h , 11235_h, ... ,	$1125A_h$, $1125B_h$	(40 Wörter)
Zweite Zeile:	$1125C_h$, $1125D_h$, ... ,	11282_h, 11283_h	(40 Wörter)
Dritte Zeile:	11284_h, 11285_h, ... ,	usw.	

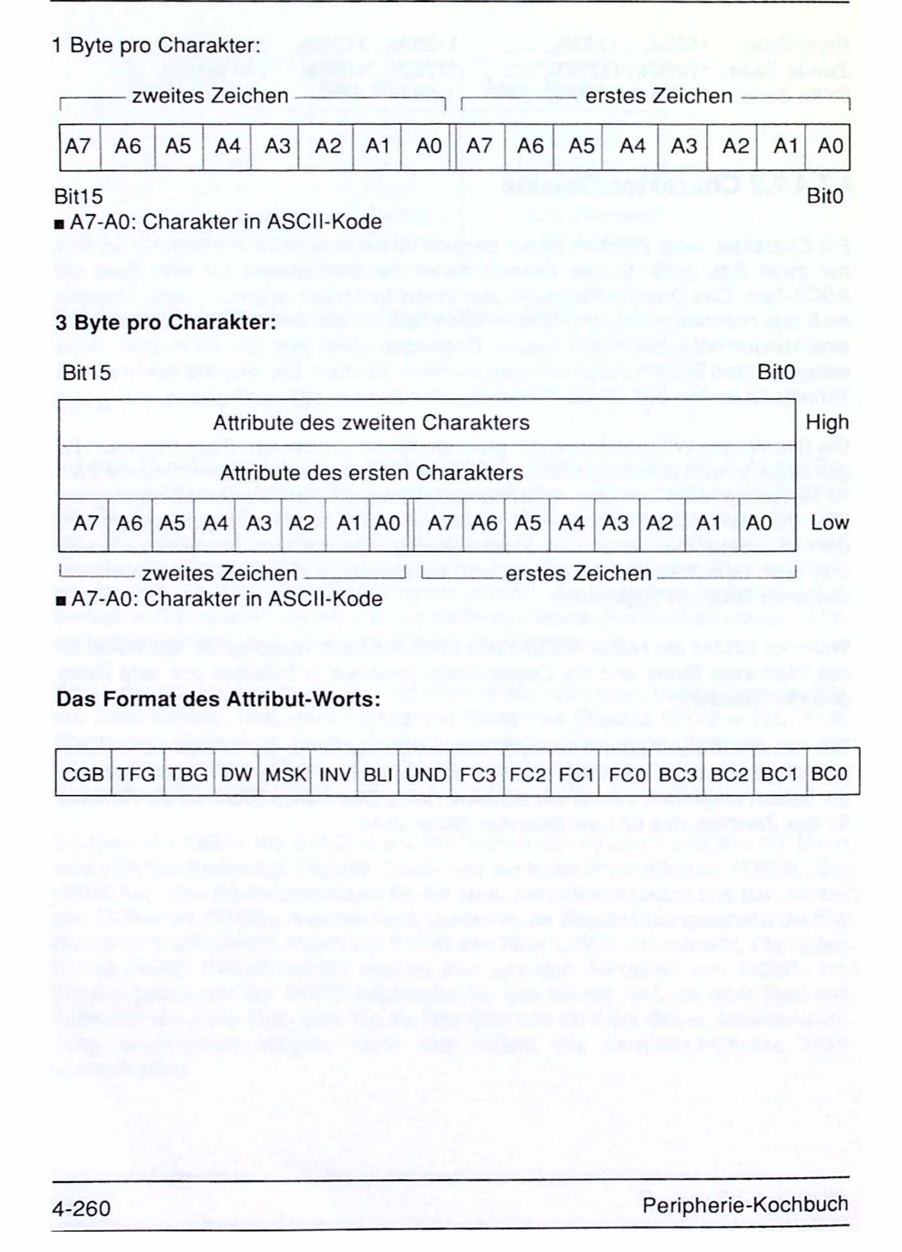
4.7.4.7.2 Charakter-Objekte

Für Charakter oder Zeichen ist der Bereich für die temporäre Adresse N0 bis N11 nur zwölf Bits groß. Dieser Bereich nennt die Startadresse für eine Zeile mit ASCII-Text. Das Objekt selbst kann aus vielen Textzeilen bestehen. Jede Textzeile muß aus mehreren Bildschirmzeilen bestehen, da die Zeichen für die Lesbarkeit eine Mindesthöhe besitzen müssen. Deswegen bildet jede Bildschirmzeile einen waagerechten Schnitt durch ein oder mehrere Zeichen. Die aktuelle Nummer des Schnitts ist in den Bits Z0 bis Z3 der Beschreibungstabelle enthalten.

Die Bits W0 bis W5 arbeiten in der gleichen Weise wie bei den Pixel-Objekten. Es gibt dabei jedoch einen signifikanten Unterschied: Ist für die Zeichenhöhe die Zahl 16 festgelegt, wird derselbe Adreßbereich 16mal für die Zeilenkonstruktion gelesen. Der mitlaufende Schnitt- oder Zeilenzähler Z0 bis Z3 wird benutzt, um aus dem Charakter-Generator den entsprechenden Schnitt des Zeichens zu holen. Erst nach dem 16fachen Durchlauf wird auf den folgenden Adreßbereich mit der nächsten Textzeile zugegriffen.

Wenn im DRAM ein reiner ASCII-Kode ohne Attribute abgelegt ist, beinhaltet jedes Wort zwei Bytes und die Objekt-Breite kann nur in Schritten von acht Bytes geändert werden.

Werden ASCII-Zeichen mit Attribute verwendet, stehen in drei Wörtern zwei Charakter. Das erste der drei Wörter beinhaltet den ASCII-Kode der beiden Zeichen, die beiden folgenden Wörter die Attribute dafür. Das zweite Wort hat die Attribute für das Zeichen, das im Low-Byte des Worts steht.



1 Byte pro Charakter:

zweites Zeichen								erstes Zeichen							
A7	A6	A5	A4	A3	A2	A1	A0	A7	A6	A5	A4	A3	A2	A1	A0

Bit15 Bit0

■ A7-A0: Charakter in ASCII-Kode

3 Byte pro Charakter:

Bit15 Bit0

Attribute des zweiten Charakters																High
Attribute des ersten Charakters																
A7	A6	A5	A4	A3	A2	A1	A0	A7	A6	A5	A4	A3	A2	A1	A0	Low
zweites Zeichen								erstes Zeichen								

■ A7-A0: Charakter in ASCII-Kode

Das Format des Attribut-Worts:

CGB	TFG	TBG	DW	MSK	INV	BLI	UND	FC3	FC2	FC1	FC0	BC3	BC2	BC1	BC0

Es bedeuten:

BC0-BC2 :	Hintergrundfarbe
BC3:	Wenn HCR=0, läßt es mit 1 die obere oder mit 0 die untere Hälfte des Zeichens in doppelter Höhe erscheinen. Wenn HCR = 1, ist es das vierte Bit der Hintergrundfarbe.
FC0-FC2 :	Farbe des Charakter-Zeichens
FC3 :	Wenn HCR = 0, läßt es mit einer 1 das Zeichen in doppelter Höhe erscheinen. Wenn HCR = 1, ist es das vierte Bit der Charakter-Farbe. Wenn HCR = 0 und FC3 = 0, muß BC3 ebenfalls auf Null gesetzt werden.
UND :	Mit einer 1 wird das Zeichen unterstrichen.
BLI :	Eine 1 läßt das Zeichen blinken.
INV :	Mit einer 1 werden Vordergrund- und Hintergrundfarbe vertauscht.
MSK :	Mit einer 1 besitzt das Zeichen dieselbe Farbe wie der Hintergrund.
DW :	Eine 1 bewirkt die Verdoppelung der Zeichenbreite.
TBG :	Mit einer 1 wird der Hintergrund des Zeichens transparent und Objekte hinter dem Zeichen sind sichtbar.
TFG :	Mit einer 1 wird das Zeichen selbst transparent und andere Objekte sind durch das Zeichen hindurch sichtbar, wenn TBG=0.
CGB :	Das Bit wählt aus zwei Charakter-Generatoren einen aus.

4.7.4.7.3 Die Charakter-Generatoren

Bei gegebenem ASCII-Kode muß der VSDD entscheiden, welcher Schnitt des Charakters anzuzeigen ist und wo er die Daten dafür holen kann. Diesen Speicherbereich, in dem die Pixel-Informationen für die darzustellenden Zeichen gespeichert sind, nennt man Charakter-Generator. Die meisten Computersysteme benutzen feste ROM-Bausteine als Charakter-Generatoren. Da sie nur lesbar sind, kann man die Zeichenform in ihnen nicht ändern.

Ein Charakter-Generator für den VSDD ist ein Speicherbereich im DRAM und besteht aus n Blöcken von 256 Wörtern, wobei mit $n \leq 16$ die Höhe eines Zeichens in Bildschirmzeilen ist. Ein jeder Block besteht aus einem Schnitt aus allen 256 Zeichen.

Der VSDD gestattet die gleichzeitige Benutzung von zwei verschiedenen Charakter-Generatoren mit je 256 Zeichen. Die Bits 12 bis 15 ihrer Basisadresse ist Inhalt des Registers R10 im Registersegment. Dadurch kann die Anfangsadresse eines Charakter-Generators nur in Schritten von 4 KWord in Bank 0 erfolgen. Es ist möglich, mehr als nur zwei Charakter-Generatoren im Bildschirmspeicher anzusiedeln, es können aber nur zwei zu derselben Zeit benutzt werden.

Wie in Bild 4-89 dargestellt, besteht die Adresse für einen Charakter-Generator aus vier Feldern.:

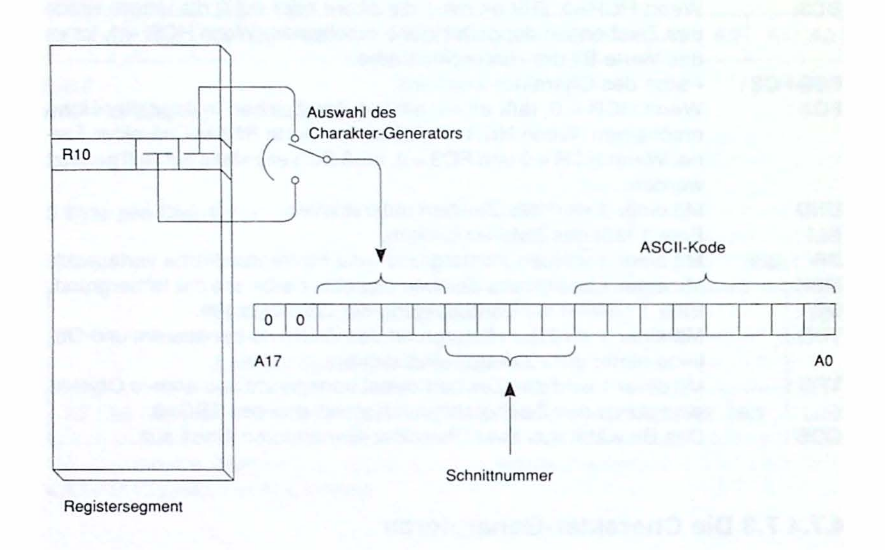


Bild 4-89. Die Adressierung des Charakter-Generators

Wenn die Zeichen ohne Attribute im DRAM abgelegt sind, wird der Charakter-Generator mit dem Bit CRS in der Objekt-Beschreibungstabelle ausgewählt. Wenn die Zeichen mit vollen Attributen versehen sind (FAD = 1), kann der Charakter-Generator für jedes Zeichen separat gewählt werden. Reine ASCII-Objekte verfügen also über 256 verschiedene Zeichen in einem Objekt, während ASCII-Objekte mit allen Attributen über 512 verschiedene Zeichen verfügen.

Jeder Schnitt eines jeden Charakters beansprucht den Platz von einem Wort. Im Wort wird der Schnitt als eine Sequenz von Einsen und Nullen abgelegt, wobei rechts und links vertauscht sind. D.h. die Eintragung im untersten Bit wird zuerst ausgelesen und auf dem Bildschirm auf der linken Seite des Zeichens dargestellt. Eine eingetragene Eins in den Charakter-Generator bewirkt die Darstellung in der Vordergrundfarbe.

Die Zeichenbreite kann entweder konstant oder proportional sein. Die Auswahl zwischen beiden Möglichkeiten erfolgt durch das Bit PSE in der Objekt-Beschreibungstabelle. Eine Null bedeutet konstanter Abstand, eine Eins Proportionalabstand.

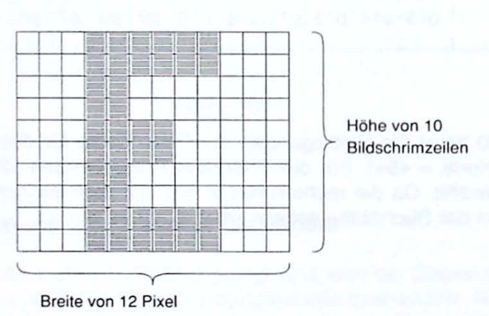
Für Zeichen mit **konstantem Abstand** kann mit den Bits R0 und R1 in der Objekt-Beschreibungstabelle eine Breite von 6, 8, 12 oder 16 Pixel gewählt werden. In diesem Fall benutzt der VSDD nur die rechtsbündigen Bits (pn) in einem Wort des Charakter-Generators. Die überzähligen Bits, die beliebigen Inhalt (x) aufweisen können, werden ignoriert:

Pixel/Char.	**Benutzte Bits in einem Charakter-Wort = pn**															
6	x	x	x	x	x	x	x	x	x	x	p5	p4	p3	p2	p1	p0
8	x	x	x	x	x	x	x	x	p7	p6	p5	p4	p3	p2	p1	p0
12	x	x	x	x	p11	p10	p9	p8	p7	p6	p5	p4	p3	p2	p1	p0
16	p15	p14	p13	p12	p11	p10	p9	p8	p7	p6	p5	p4	p3	p2	p1	p0

Bild 4-90 zeigt die Eintragungen der Pixel-Werte für das Charakter-Zeichen "E" (ASCII-Kode = 45_h). Für die Charakter-Breite wurden 12 Pixel, für die Höhe 10 Pixel gewählt. Da die rechten Bits zuerst auf dem Bildschirm dargestellt werden, erscheint der Buchstabe seitenverkehrt.

Adresse im Char.-Generator	Pixel-Speicher P11 .... P0	Hex-Kode
01945	0 0 0 0 0 0 0 1 1 1 1 1 1 0 0 0	01F8
01845	0 0 0 0 0 0 0 1 1 1 1 1 1 0 0 0	01F8
01745	0 0 0 0 0 0 0 0 0 0 0 1 1 0 0 0	0018
01645	0 0 0 0 0 0 0 0 0 0 0 1 1 0 0 0	0018
01545	0 0 0 0 0 0 0 0 0 1 1 1 1 0 0 0	0078
01445	0 0 0 0 0 0 0 0 0 1 1 1 1 0 0 0	0078
01345	0 0 0 0 0 0 0 0 0 0 0 1 1 0 0 0	0018
01245	0 0 0 0 0 0 0 0 0 0 0 1 1 0 0 0	0018
01145	0 0 0 0 0 0 0 1 1 1 1 1 1 0 0 0	01F8
01045	0 0 0 0 0 0 0 1 1 1 1 1 1 0 0 0	01F8

a. Bitspeicher für das Zeichen "E"



b. Das Zeichen "E" auf dem Bildschirm

Bild 4-90. Die Daten für das Zeichen "E" im Charakter-Generator

Mit dem Setzten des Bits PSE in der Objekt-Beschreibungstabelle wird der **Proportionalabstand** der Zeichen gewählt. Dabei kann für ein Zeichen unabhängig vom anderen eine Breite von 2, 4, 6, 8, 10, 12 oder 14 Pixel gewählt werden, die von der Art des Zeichens abhängt. Die Information über die Breite entnimmt der VSDD den drei höchsten Bits (den zwei höchsten Bits bei der Breite von 14).

Pixel/Char.	Breite			Pixel pn												
2	0	0	0	x	x	x	x	x	x	x	x	x	x	x	p1	p0
4	0	0	1	x	x	x	x	x	x	x	x	x	p3	p2	p1	p0
6	0	1	0	x	x	x	x	x	x	x	p5	p4	p3	p2	p1	p0
8	0	1	1	x	x	x	x	x	p7	p6	p5	p4	p3	p2	p1	p0
10	1	0	0	x	x	x	p9	p8	p7	p6	p5	p4	p3	p2	p1	p0
12	1	0	1	x	p11	p10	p9	p8	p7	p6	p5	p4	p3	p2	p1	p0
14	1	1	p13	p12	p11	p10	p9	p8	p7	p6	p5	p4	p3	p2	p1	p0

4.7.4.8 Die Farbtabelle

Mit Register R9 im Registersegment wird die Basisadresse der Farbtabelle festgelegt. Die Farbtabelle ist ein Bereich in Bank 0 des Bildschirmspeichers von 16 Wörtern, den der VSDD zu Beginn eines jeden Bildaufbaus komplett liest. Im Speicher können mehrere Farbtabellen gleichzeitig stehen. Durch Ändern der Basisadresse kann bequem auf eine andere Tabelle umgeschaltet werden. Somit können die Farben von Bild zu Bild vollständig gewechselt werden.

Jedes der 16 Wörter in der Farbtabelle wird in vier Felder eingeteilt, wovon drei die Grundfarben und eines den Farbindex in der Tabelle aufweist:

R3	R2	R1	R0	G3	G2	G1	G0	B3	B2	B1	B0	A3	A2	A1	A0

Die folgende Tabelle 4-48 stellt ein Beispiel für eine übliche Farbprogrammierung dar. Die Farbtabelle beginnt in diesem Beispiel an der DRAM-Adresse 10, daraus folgt die Adresse 20 für den Prozessor.

µP-Adresse	Daten	Farbnr.	Farbe
0020	0F00	0	Grün
0022	F001	1	Rot
0024	00F2	2	Blau (Hintergrundfarbe)
0026	FFF3	3	Weiß
0028	0004	4	Schwarz
002A	FF05	5	Gelb
002C	0FF6	6	Türkis
002E	F0F7	7	Ocker
0030	AAA8	8	Grau
0032	00A9	9	Dunkelblau
0034	0A0A	10	Dunkelgrün
0036	A00B	11	Dunkelrot
0038	CFFC	12	Purpur
003A	9F5D	13	Hellgrün
003C	AFEE	14	Hellblau
003E	FA8F	15	Rosa

Tabelle 4-48. Beispiel für die Farbprogrammierung

4.7.5 Der Zwillings-Modus

Im Zwillings-Modus sind zwei 82716 parallel verbunden, um die Geschwindigkeit des Systems wachsen zu lassen. Dabei muß einer als Slave, der andere als Master definiert sein. Das erfolgt mit den Bits TMM und TMS im Register R1. Jeder Baustein konstruiert abwechselnd eine Bildschirmzeile, so daß für eine Zeilenkonstruktion die doppelte Zeit zur Verfügung steht.

In der Hardware muß die Existenz zweier VSDDs berücksichtigt werden. Obwohl jeder Chip für sich über einen eigenen Bildschirmspeicher, über einen eigenen Takt und Dekodierlogik verfügt, muß der Videotakt an beide gleichzeitig über den Eingang CKIO geführt werden. Das gewährleistet den Gleichlauf beider Synchronisierungsgeneratoren. Eine typische Anordnung zeigt Bild 4-91.

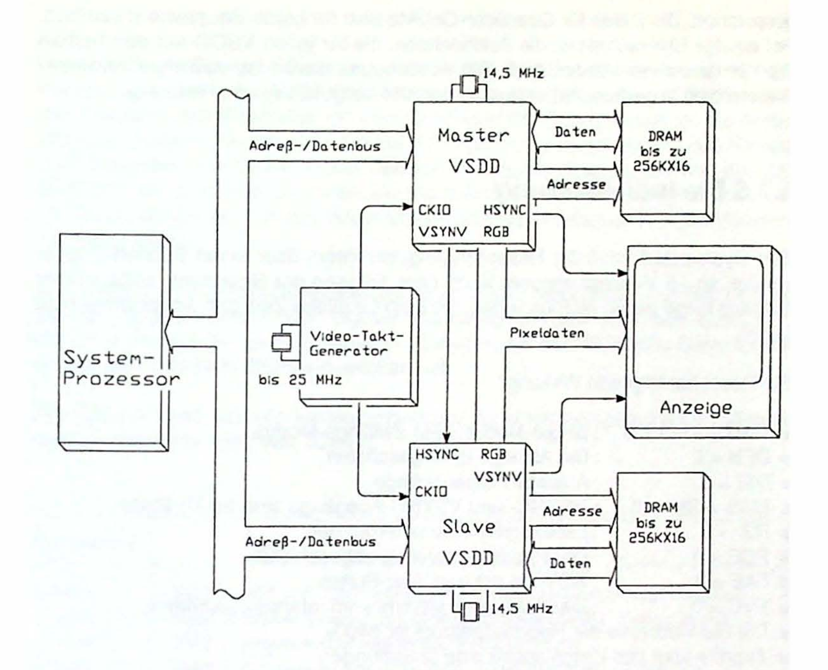


Bild 4-91. Der Anschluß zweier 82716 im Zwillings-Modus

Näheres über die Programmierung und Signalformen findet sich in Abschnitt 4.7.3.4.7.

Hinsichtlich der Software gibt es einige Aspekte, denen besondere Aufmerksamkeit zu schenken ist. Jeder Baustein muß mit Ausnahme der Bits TMM und TMS in derselben Weise konfiguriert werden. Da jeder VSDD nur die halbe Zeilenzahl anzeigt, ist die Zugriffstabelle nur halb so groß. Das hat zur Folge, daß die Objekte nur in 2-Zeilen-Schritte in senkrechter Richtung bewegt werden können. Der Inhalt beider Tabellen ist identisch.

Ferner müssen die Objekt-Beschreibungstabellen identisch sein. Die Daten für Pixel-Objekte müssen in gerade und ungerade Bildschirmzeilen aufgeteilt sein. Diese werden gesondert im ungeraden VSDD (Master) und im geraden VSDD (Slave)

gespeichert. Die Daten für Charakter-Objekte sind für beide Bausteine dieselben. Der einzige Unterschied ist die Zeichenhöhe, die für jeden VSDD auf den halben Wert programmiert werden muß. Die Konsequenz davon ist, daß die Charakter-Generatoren in gerade und ungerade Schnitte aufgeteilt werden müssen.

4.7.6 Die Initialisierung

Zum Systemstart muß der Reset-Eingang, der intern über einen Schmitt-Trigger verfügt, an +5 V gelegt werden. Nach dem Anlegen der Spannung sollte dieser Eingang lange genug an Plus liegen, um dem Oszillator Zeit zum Anschwingen zu geben.

Ein Reset hat folgende Wirkung:

- TMM = TMS = 0 : Single-Modus, kein Zwillings-Modus
- DEN = 0 : Die Anzeige ist abgeschaltet.
- DEI = 0 : Analoge Farbausgänge
- MAS = SM = 0 : HSYNC- und VSYNC-Ausgänge sind im Tri-State.
- RE = 0 : Lesezugriffe sind unterbunden.
- PCE = 0 : Der Prioritätszähler ist abgeschaltet.
- FAE = 0 : RDY-Pin erzeugt Wait-States.
- EVC = 0 : Das Taktsignal stammt vom internen Oszillator.
- Die Basisadresse der Registerfensters ist 0400_h.
- Zugriffe über das Datenfenster sind unterbunden

Der Baustein verläßt den Reset mit folgenden Standardwerten für die Bildschirm-konstanten: (Die Zeiten gelten für einen 14,5-MHz-Quarz)

- HC0 = 1: HSYNC-Breite = 32•Taktperiode (2,2 µs)
- HC1 = 2: Startzeit = 48•Taktperiode (3,3 µs)
- HC2 = 5: Stoppzeit = 96•Taktperiode (6,6 µs)
- HC3 = 8: HSYNC-Periode = 144•Taktperiode (9,9 µs)
- VC0 = 1: VSYNC-Breite = 2 Zeilen (19,8 µs)
- VC1 = 2: Startzeit = 3 Zeilen (29,7 µs)
- VC2 = 3: Stoppzeit = 4 Zeilen (39,6 µs)
- VC3 = 7: VSYNC-Breite = 8 Zeilen (79,2 µs)

Mit diesen Zeilen ist natürlich kein Bildaufbau möglich. Sie haben den Hauptzweck, keine ungültigen Zufallswerte für die Steuerung des Bildschirms zuzulassen.

Unmittelbar nach einem Reset führt der VSDD acht Refresh-Zyklen auf den gesamten Speicher aus. Die CPU darf daher frühestens 10 µs (bei 14,5 MHz) nach der Freigabe des Bausteins mit dem Schreiben der Daten beginnen. Die ersten Daten müssen in die Register R0 bis R15 fließen, da durch sie erst das Fenster zum Schreiben der Objekt-Daten definiert wird. Es ist darauf zu achten, daß das UCF-Bit zunächst gelöscht bleibt, bis alle Initialisierungsdaten komplett im VSDD vorliegen, da ein Setzten des Bits eventuell das Verschieben des Registerfensters zur Folge hätte.

Nach jedem kompletten Bildaufbau testet der VSDD den Zustand des Bits UCF. Ist es auf 1 gesetzt, liest er alle Register und konfiguriert sich damit neu. Somit muß die CPU die Zeit für einen Bildaufbau abwarten, bis der VSDD das Datenfenster an der gewünschten Stelle eingerichtet hat.

Die Tabelle 4-49 gibt ein Initiierungsbeispiel für einen europäischen TV-Monitor und einem Systemtakt von 10 MHz.

µP-Adresse		Daten Wirkung
0400	5403	UCF und DEN werden gelöscht; 640 Zeilenpixel; analoge Ausgänge
0402	860D	Interner Videotakt bei 10 MHz; Composite-Modus; Pipeline-Modus
0404	7FF6	Registerfenster beginnt bei FFFE0; kein Rand
0406	00D8	Datenfenster beginnt bei 0000; Bildschirmgrenze = 216•2 Pixel
0408	C000	Datenfensterbreite 32 KByte
040A	0000	Datensegment beginnt bei 0 im DRAM (es überlappt die Register!)
040C	0000	Keine Limitierung der Zugriffe
040E	0040	Objekt-Beschreibungstabelle beginnt an Adresse 40 im DRAM
0410	0200	Basisadresse der Zugriffstabelle ist 200
0412	0010	Basisadresse der Farbtabelle
0414	0089	Basisadressen der Charakter-Generatoren 8000 und 9000
0416	----	wird nicht beschrieben
0418	0400	32 Pixel bzw. 1 Zeile
041A	1C27	112 Pixel bzw. 40 Zeilen
041C	8921	544 Pixel bzw. 290 Zeilen
041E	9D37	640 Pixel bzw. 312 Zeilen

Tabelle 4-49. Beispiel von Initiierungsdaten

4.7.7 Die Anzeigenleistung

Die Zeit, die der VSDD für den Aufbau einer Bildschirmzeile hat, ist die Zeit der horizontalen Periode des Elektronenstrahls auf dem Bildschirm. In einem standardmäßigen TV-Gerät beträgt diese Zeit ungefähr 64 µs. In dieser Zeit muß der 82716 feststellen, welche Objekte in der Zeile erscheinen sollen, den Objekt-Typ und die Adresse der Objekt-Daten. Die Daten müssen daraufhin gelesen und in den Zeilenspeicher geschrieben werden.

Da jedes aktive Objekt eine Serie von Speicherzugriffen mit sich zieht, dauert es natürlich um so länger, je größer die Zahl der darzustellenden Objekten ist. Ein Maß der Anzeigenleistung ist die Zahl der Objekte, die in derselben Zeile darstellbar ist. Da die Zahl auch von der Objekt-Breite abhängt, ist ein genaueres Maß für die Leistung die Zahl der behandelten Pixel pro Zeile.

Eine Rolle spielt auch der Systemtakt und die Geschwindigkeit der Speicherbausteine. Langsame Bausteine haben eine Zugriffszeit von 210 ns, schnelle DRAMs haben eine Zugriffszeit von 140 ns. In den meisten Fällen ist ein Takt von 14,5 MHz empfehlenswert, der eine Taktperiode von 70 ns besitzt. Auf diese Zahlen nehmen die folgenden Betrachtungen Bezug.

Die Gesamtzeit, die dem VSDD zur Pixel-Verarbeitung zur Verfügung steht ist:

t = Zeit einer Bildschirmzeile - Zeilenauflage - (Objekt-Zahl•Objekt-Auflage)

Die Zeit für eine Bildschirmzeile ist 64 µs. Unter der Zeilenauflage versteht man alle Verpflichtungen, die der VSDD zwischen zwei Zeilen hat. Da ist vor allem der Block-Refresh und verschiedeen andere Reinitialisierungen. Dafür werden 86 Taktperioden verwendet mit einer Dauer von 6 µs bei schnellen und 6,7 µs bei langsamen DRAMs. Ähnlich ist die Objekt-Auflage zu Verstehen. Das sind Auflagen, die ein Objekt dem VSDD macht, bis er die Daten aus dem Speicher lesen kann. Die Zeiten hierfür sind 3,1 bzw. 3,4 µs pro Objekt. Nach Abrechnung des Zeitbedarfs bleiben bei einer Darstellung von vier Objekten für die eigentlichen Pixel eine Zeit von 45,6 µs bzw. 43,7 µs. Der VSDD kann in dieser Zeit 325mal aus schnellen DRAMs und 208mal aus langsamen DRAMs lesen. Je nach gewählter Bildauflösung werden bei jedem Speicherzugriff 2, 4 oder 8 Pixel geholt, so daß während einer Zeilenkonstruktion 650, 1300 oder 2600 Pixel aus schnellen RAMs gelesen werden können.

Die Zahl der Pixel, die der VSDD für eine Zeilenkonstruktion lesen muß, kann durchaus eine größere sein, als auf dem Bildschirm angezeigt wird. Das hat seinen Grund darin, daß Objekte niedrigerer Priorität von Objekten höherer Priorität überschrieben werden.

Die Zeit, die der VSDD zur Darstellung von Charakter-Zeichen benötigt, ist eine andere als bei Pixel-Objekten. Mit einem Wort nimmt der VSDD die Kodes zweier ASCII-Zeichen auf. Er muß mit Hilfe der Schnittnummer auf den Charakter-Generator zugreifen, um die eigentlich darzustellenden Pixel zu erhalten. Etwas mehr Zeit benötigt er, wenn zu jedem Zeichen noch die Attribute berücksichtigt werden müssen. Die nachstehende Beziehung geben die Zeiten unter den ungünstigsten Bedingungen an (Die Zahlen in Klammern beziehen sich auf langsame DRAM-Bausteine):

Für reine ASCII-Objekte (1 Byte pro Charakter):

$$\text{Zahl der Taktperioden} = \frac{\text{Zahl der Charakter}}{2} \bullet 14\ (17) + 1$$

Für ASCII-Objekte mit Attributen (3 Byte pro Charakter):

$$\text{Zahl der Taktperioden} = \frac{\text{Zahl der Charakter}}{2} \bullet 16\ (19) + 1$$

Frequenzbereiche:

Der VSDD ist in drei Frequenzbereichen erhältlich, die durch eine anghängte Zahl an der Bausteinnummer zu erkennen sind:

82716-3 10,0 MHz
20 MHz Videotakt

82716-4 11,0 MHz
22 MHz Videotakt

82716-5 13,3 MHz
22 MHz Videotakt

Literaturverzeichnis

- J. Elsing/A. Wiencek: Schnittstellenhandbuch. IWT Verlag, Vaterstetten, 1987
- Wohak B.: 8086 286 Assembler. IWT Verlag, Vaterstetten, 1988
- Roth A.: Das Mikrocontroller-Kochbuch. IWT Verlag, Vaterstetten, 1989
- Bernstein H.: Hardware-Handbuch PC-XT-AT und Kompatible. Markt&Technik Verlag, 1988
- Sridhar Begur: 82C08 User's Manual. Intel, 1986
- Viren Shah: A Low Cost and High Integration Graphics System Using 82716. Intel, AP-268, 1986
- Intel: Microprozessor and Peripheral Handbook. Volume I, Volume II, 1987
- Intel: Embedded Control Applications. 1988
- Shafer Michael A.: 8051 Based CRT Terminal Controller. Intel, AP-223, 1984
- Jigour Robin: Using the 8259A Programmable Interrupt-Controller. Intel, AP-59, 1986
- Alexy George: 8086 System Design. Intel, AP-67, 1986
- Hitachi: Application of Dynamic RAMs. 1985
- Harris: Digital. Data Book, 1988
- MHS: Digital, Catalog and Asic Products. 1987
- MHS: VSDD 82716 User's Manual
- David Bell: Video Storage and Display Device (VSDD). MHS, AN-1030, 1987
- Mitsubishi: Microprozessors and Peripheral Circuits. 1987
- Laurence E. Laumann: D.A.T.A.Book, Microprozessor Integrated Circuits. 1987

Stichwortverzeichnis

C

D

E

F

G

H

I

K

L

M

N

O

P

R

S

T

U

V

W

Z

Das CMOS-Kochbuch

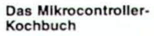
Die digitale CMOS-Bausteinserie ist eine der modernsten und zukunftssichersten Logikfamilien. Hier liegt die umfassendste neutrale Darstellung darüber vor. Das CMOS-Kochbuch bringt eine Fülle von Informationen, die es schnell zu einem unentbehrlichen Ratgeber für Elektroniker machen wird.
Übersetzung aus dem Amerikanischen.

1989. 5. über. Aufl. 520 Seiten. Geb.
ISBN 3-88322-002-7
DM 68,-

Das Aktiv-Filter-Kochbuch

Das Aktiv-Filter-Kochbuch stellt ein praktisches, anwenderorientiertes Nachschlagewerk für jeden dar, der etwas über den Aufbau eines speziellen Filters wissen will. Der Name „Lancaster", auch Autor des CMOS-Kochbuches, steht für die Qualität dieses Werkes.
Übersetzung aus dem Amerikanischen.

1989. 5. Aufl. 276 Seiten. Geb.
ISBN 3-88322-007-8
DM 48,-



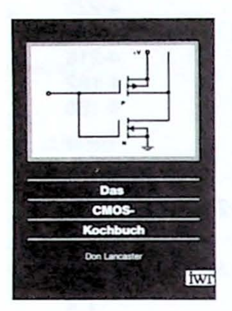
Das Mikrocontroller-Kochbuch

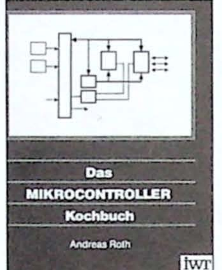
Dieses Buch beschreibt exemplarisch den Baustein 8031 als Vertreter der MCS-51-Familie und setzt den Schwerpunkt in der praktischen, anwenderorientierten Darstellung des Stoffes. Es ist gleichermaßen für Einsteiger und Praktiker geeignet.

1990. 2. Aufl. 392 Seiten. Geb.
ISBN 3-88322-225-9
DM 58,-

TTL-Taschenbuch, Teil 2 (74201 — 74640)

1987. 4. Aufl. 324 S. Kart.
ISBN 3-88322-192-9
DM 32.-















Angewandte Mikroelektronik, Band 1

Das Buch bietet eine praxisorientierte Einführung in die Mikroelektronik. Band 1 behandelt neben den notwendigen Grundlagen die Themen Operationsverstärker, Sensoren, Grundlagen der Digitaltechnik, Analog-Digital-Wandler und Digital-Analog-Wandler. Der Leser wird durch eine Reihe von ausgearbeiteten Experimenten, für die auch gedruckte Schaltungen erhältlich sind, unterstützt.

1990. 312 Seiten. Geb.
ISBN 3-88322-283-6
DM 68,-

Angewandte Mikroelektronik, Band 2

Dieser Band bietet eine praxisorientierte Einführung in die Mikroprozessortechnik. Es werden alle Elemente von mikroprozessorgesteuerten Schaltungen — Speicher, Prozessor, Peripherie — besprochen und anhand von ausgearbeiteten Experimenten dem Leser vermittelt. Zur Unterstützung des Lesers sind gedruckte Schaltungen erhältlich, die den Einstieg in dieses komplexe Gebiet erleichtern.

1990. 368 Seiten. Geb.
ISBN 3-88322-284-4
DM 68,-

TTL-Taschenbuch, Teil 1 (74oo — 74200)

Die Taschenbücher bieten eine klar gegliederte Zusammenstellung aller gängigen TTL-Bausteine. Es sind die Produkte aller namhaften Hersteller erfaßt und in einer übersichtlichen Matrix am Ende jedes Teils zusammengestellt.

1990. 4. Aufl. 312 Seiten. Kart.
ISBN 3-88322-191-0
DM 32,-

TTL-Taschenbuch, Teil 3 (74641-7430640)

1987. 4. Aufl. 300 S. Kart.
ISBN 3-88322-193-7
DM 32.-

Über das Buch:

Das Computer-Peripherie-Kochbuch konzentriert sich auf die praxisnahe Beschreibung der Bausteine, ohne die auch ein Super-Prozessor hilflos wäre, und zeigt mit zahlreichen Hardware-Beispielen deren praktische Verwendung und Programmierung. Im Vordergrund der Betrachtung stehen die Bausteine der 82XX-Reihe, die kompatibel zu den Mikrocontrollern und Prozessoren der 80XX-Reihe sind.

Nach einer globalen Einführung in 8-Bit- sowie 16-Bit-Busstrukturen und DRAM-Konfigurationen werden neben vielen anderen folgende Bausteine behandelt:

- *Taktgeneratoren 8284, 82284, 82384*
- *Buscontroller 8288, 82288*
- *Busverwalter 8289*
- *Programmierbare Chip-Select-Dekoder 82C138, 82C139, 82C338, 82C339*
- *Interrupt-Controller 8259*
- *Intervall-Timer-Baustein 8253*
- *DMA-Controller 8237*
- *Controller für dynamische RAMs 82C08*
- *I/O-Portbaustein 8255*
- *Kommunikationsbaustein 8250*
- *Videospeicher- und Anzeigebaustein (VSDD) 82716*

Zur Erleichterung von Eigenentwicklungen sind die Pin-Belegungen und eine kurze Beschreibung der Mikrocontroller MCS-48, MCS-51, MCS-96, der Prozessoren 8080/85, 8086/88, 80186/188, 80286, 80386 und der Coprozessoren 8087 und 80287 aufgeführt.

Die RS-232-C-Schnittstelle, Stromausfallerkennung, Monitoransteuerung, Einblendung von Computer-Grafiken in Videoaufnahmen, Multiprozessorensysteme sind nur ein winziger Ausschnitt aus zahlreichen Hardware-Realisationen.

Bei der Konzeption des Buches wurde größter Wert auf Praxisnähe gelegt. Dies äußert sich in einer erschöpfenden Darstellung der Bausteineigenschaften und ihrer Zusammenarbeit mit der CPU. 208 Abbildungen mit Pin-Belegungen, Blockdiagrammen und Schaltplänen sowie zahlreiche Tabellen gestatten nicht nur die Entwicklung eines eigenen leistungsfähigen Computersystems, sondern helfen durch Querverweise auf IBM-kompatible Systeme beim Verständnis und Studium von Hardware-Strukturen und erleichtern nicht zuletzt die Systemprogrammierung.

Über den Autor:

Andreas Roth wurde 1952 geboren und beschäftigt sich seit 1974 in Lehre und Entwicklung mit der digitalen Halbleiterelektronik. Gute Verbindungen zu namhaften Halbleiterherstellern trugen zur Entstehung des vorliegenden Buches bei.

Bereits erschienen:
Das Mikrocontroller-Kochbuch

ISB N 3-88322-234-8

DM 68,-

9 783883 222349

Hardware

4.7.8 Entwicklungshilfen

Wegen der Komplexität des Video-Controllers 82716 stellen die Hersteller des Bausteins Hilfen in Form von Soft- und Hardware zur Verfügung.

Vom Hersteller Matra-Harris Semiconductor (MHS) existieren für verschiede Prozessorensysteme Entwicklungs-Kits:

Die Kontaktadresse für Deutschland ist:

MHS
Erfurterstraße 29
D-8057 Eching

VDT 716:
Das VSDD Development Tool (VDT) besteht aus zwei Komponenten, dem Video Development Board (VDB) und der Video Development Software (VDS).

Das VDB 716 ist eine bestückte Platine, die so geschaffen ist, daß sie in den XT-Slot eines IBM-kompatiblen Personal Computers paßt. Auf der Steckkarte befindet sich neben der erforderlichen Interface-Logik ein 8086-Prozessor mit einem 256 KByte EPROM, der VSDD 82716 mit 512 KByte Bildschirmspeicher, ein UART 8251, Interruptcontroller 8259, Timer 8254 mit Maus für die Grafikbedienung. Leersockel gestatten die Erweiterung auf den Zwillings-Modus und die Verwendung einer externen Farbtabelle.

Das VDS 716 ist die Software zur Bedienung des VDB 716 auf zwei 5-Zoll-Disketten. Diese Programm gestattet die Eingabe verschiedener Parameter wie Initiierungsdaten, Charakter-Generatoren, Farbtabellen etc. Ferner ist der Aufbau und die Bewegung von Bildern möglich, ein komfortabler Befehlssatz steht zu diesem Zweck zur Verfügung. Der gesamte Inhalt des Bildschirmspeichers kann auf Diskette abgelegt werden oder von Diskette neu geladen werden. Eine Dokumentation ist beigefügt.

4.7.8 Entwicklungshilfen

Wegen der Komplexität des Video-Controllers 82716 stellen die Hersteller des Bausteins Hilfen in Form von Soft- und Hardware zur Verfügung.

Vom Hersteller Matra-Harris Semiconductor (MHS) existieren für verschiede Prozessorensysteme Entwicklungs-Kits:

Die Kontaktadresse für Deutschland ist:

MHS
Erfurterstraße 29
D-8057 Eching

VDT 716:
Das VSDD Development Tool (VDT) besteht aus zwei Komponenten, dem Video Development Board (VDB) und der Video Development Software (VDS).

Das VDB 716 ist eine bestückte Platine, die so geschaffen ist, daß sie in den XT-Slot eines IBM-kompatiblen Personal Computers paßt. Auf der Steckkarte befindet sich neben der erforderlichen Interface-Logik ein 8086-Prozessor mit einem 256 KByte EPROM, der VSDD 82716 mit 512 KByte Bildschirmspeicher, ein UART 8251, Interruptcontroller 8259, Timer 8254 mit Maus für die Grafikbedienung. Leersockel gestatten die Erweiterung auf den Zwillings-Modus und die Verwendung einer externen Farbtabelle.

Das VDS 716 ist die Software zur Bedienung des VDB 716 auf zwei 5-Zoll-Disketten. Diese Programm gestattet die Eingabe verschiedener Parameter wie Initiierungsdaten, Charakter-Generatoren, Farbtabellen etc. Ferner ist der Aufbau und die Bewegung von Bildern möglich, ein komfortabler Befehlssatz steht zu diesem Zweck zur Verfügung. Der gesamte Inhalt des Bildschirmspeichers kann auf Diskette abgelegt werden oder von Diskette neu geladen werden. Eine Dokumentation ist beigefügt.